# VGUS 串口屏用户开发指南

# 修 改 记 录

| 日期 | 修改内容 |
| --- | --- |
| 2015-6-1 | 初稿。 |
| 2015-8-15 | 增加音频播放功能。<br>增加视频播放功能。<br>增加二维码显示功能。 |
| 2016-4-7 | 增加文本滚动显示功能。 |
| 2017-06-20 | 更新第 1、2 章内容。<br>更新第 4、5 章次序。<br>增加第 7 章“VGUS 开发工具使用说明”及虚拟串口屏。 |
| 2018-11-30 | 增加 1.2 节硬件连接<br>增加 4.4 节 RTC 时钟控件时钟源选择功能<br>增加 4.5 节视频控件功能<br>增加附 1 SDWa 系列功能描述<br>增加附 2 系统自检寄存器 0xf3-0xf6 描述<br>修改第 7 章内容 VGUS 开发工具使用说明 |
| 2019-08-21 | 修订超链接地址<br>修改第 7 章内容 VGUS 开发工具使用说明 |
| 2020-04-17 | 增加产品分类特点介绍 1.2 节<br>在屏参配置 7.2 节中增加选择触摸屏类型<br>删除屏参配置读写的描述<br>删除在线下载的描述<br>增加 SDWn 系列的相关描述 |

# 目　录

**1 VGUS 组态串口屏概述 ..... 5**
1.1 特点介绍 ..... 5
1.2 产品分类 ..... 6
1.3 认识组态 ..... 7
1.4 存储空间 ..... 9
**2 串口指令集 ..... 10**
2.1 格式约定 ..... 10
2.2 寄存器读写指令 0x80、0x81 ..... 11
2.3 变量存储器读写指令 0x82、0x83 ..... 12
2.4 扩展指令 0x85 ..... 13
2.5 CRC 校验 ..... 14
**3 寄存器 ..... 16**
3.1 寄存器一览表 ..... 16
3.2 常用寄存器 ..... 19
3.2.1 读取版本号 ..... 19
3.2.2 修改背光亮度 ..... 19
3.2.3 控制蜂鸣器 ..... 19
3.2.4 切换图片 ..... 19
3.2.5 按键触发 ..... 19
3.2.6 时钟 RTC ..... 20
3.2.7 触摸屏校准 ..... 20
3.2.9 语音播放寄存器 ..... 21
3.2.10 视频播放寄存器 ..... 22
3.2.11 用户数据文件 ..... 23
3.2.12 数据库读写 ..... 24
**4 显示控件 ..... 25**
4.0 显示控件介绍 ..... 25
4.1 数据与文本变量 ..... 27
4.1.1 数据变量 ..... 27
4.1.2 文本变量 ..... 28
4.1.3 滚动文本 ..... 29
4.2 图片与图标 ..... 30
4.2.1 变量图标 ..... 30
4.2.2 动画图标 ..... 31
4.2.3 动画图片 ..... 32
4.2.4 滑动刻度 ..... 33
4.2.5 艺术字变量 ..... 34

4.2.6 旋转图标 ..... 35
4.2.7 位变量图标 ..... 36
4.3 图形变量 ..... 38
4.3.1 实时曲线（趋势图） ..... 38
4.3.2 基本图形（绘图） ..... 40
4.3.3 列表显示（表格） ..... 44
4.3.4 二维码 ..... 45
4.4 实时时钟 RTC 的显示 ..... 46
4.5 视频控件 ..... 49
**5 触摸控件 ..... 50**
5.0 触摸控件介绍 ..... 50
5.1 基本按钮 ..... 52
5.1.1 按钮 ..... 52
5.1.2 按钮键值返回 ..... 53
5.1.3 按钮状态返回 ..... 53
5.2 弹出菜单 ..... 55
5.3 键盘录入 ..... 57
5.3.1 制作键盘 ..... 57
5.3.2 数据录入 ..... 58
5.3.3 英文 ASCII 录入 ..... 60
5.3.4 中文 GBK 录入 ..... 61
5.3.5 时钟设置 ..... 63
5.4 调节控件 ..... 64
5.4.1 增量调节 ..... 64
5.4.2 拖动调节 ..... 65
5.4.3 转动调节 ..... 66
5.4.4 硬件参数配置 ..... 66
**6 描述指针 ..... 68**
**7 VGUS 开发工具使用说明 ..... 81**
7.1 操作界面 ..... 81
7.2 屏参配置 ..... 83
7.3 设计流程 ..... 85
7.4 工程下载 ..... 87
7.5 文件格式 ..... 88
**附 1 系统自检寄存器 ..... 91**
**附 2 SDWE 系列数据加密 ..... 92**
**附 3 固件更新 ..... 92**

# 1 VGUS 组态串口屏概述

## 1.1 特点介绍

- ✧ 8M-128M字节大容量Flash存储器；
- ✧ 支持多语种字库、图片、二维码显示；
- ✧ 支持硬件JPG解码，存储更高效、显示更快速；
- ✧ 支持音频、视频播放（SDWn系列不支持）；
- ✧ 支持背光亮度调节、自动待机屏保功能；
- ✧ 支持U盘脱机批量下载，有效提高下载效率、降低对操作人员要求（SDWn系列用TF卡）；
- ✧ 支持软件90度、180度、270度屏幕旋转，调整合适的可视角度；
- ✧ 支持虚拟串口屏功能，让调试更轻松、让学习更简单；
- ✧ 行业领先的电磁辐射指标，助您轻松应对ClassB；
- ✧ 配套提供操作简单的VGUS组态开发工具软件！

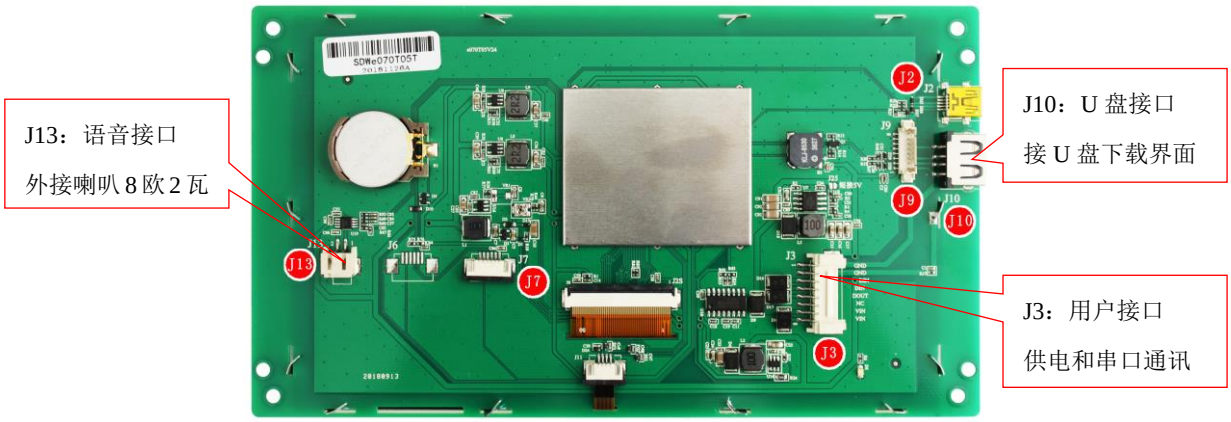



图 1-1-1 SDWe070T05T 实物照片

表 1-1-1 VGUS 串口屏外部接口汇总表

| 位号 | 名称 | 规格 | 功能 | 说明 |
|---|---|---|---|---|
| J3 | 用户接口 | SMD2.0-8 带锁 | 连接用户主板用于供电及串口通讯 | |
| J10 | U 盘接口 | USB-A 型 | 连接 U 盘用于脱机下载界面 | |
| J13 | 语音接口 | PH2.0-2 | 连接喇叭 8 欧 2 瓦 | 部分型号没有 |

关于硬件连接的详细信息请参考文档《硬件连接与串口通讯测试》。

## 1.2 产品分类

本文是以SDWe系列为例描述的，SDWa、SDWn在部分功能和性能上会有差异，详细信息参见下表。

表 1-2-1 产品分类差异汇总

| 项目 | SDWe | SDWa+ | SDWa | SDWn |
|---|---|---|---|---|
| 开发工具 | VGUS2020 | | | |
| 支持尺寸 | 15 寸~4.3 寸 | 15 寸~5 寸 | 7 寸~2.8 寸 | 3.5 寸~1.8 寸 |
| 下载文件夹 | VT_SET | VT_SET_SDWa | | VT_SET_SDWn |
| 下载方式 | U 盘 | | | TF 卡/不限容量 |
| FLASH 内存 | 128M 字节 | 16M 字节 | 8M/16M 字节 | 8M 字节 |
| 用户内存空间 | 114M 字节 | 15.1M 字节 | 7.1M /15.1M 字节 | 7.8M 字节 |
| 变量存储器 | 128K 字节 | | | 2K 字节 |
| 数据库 | 未限制 | 64K 字节 | | 不支持 |
| WAV/MP3 音频 | 2M 字节 /未限制 | 2M /2M 字节 | | 不支持 |
| AVI 视频 | 未限制 | 2M 字节 | | 不支持 |
| 显示与触摸控件 | 支持 | | | 部分不支持[1] |
| 描述指针 | 支持 | | | 不支持 |
| 寄存器 | 支持 | 不支持 0x60-0x6e | | 部分不支持[2] |
| 串口下载指令 8501 | 支持 | 不支持 | | |

***[1]详细信息参见表4-0-1、表5-0-1；***

***[2]详细信息参加表3-1-1。***

设计注意事项：

1. SDWn系列串口屏必须用最新版组态开发工具VGUS2020；
2. 新建工程时产品系列必须正确选择，根据所用产品实际型号选择SDWn、SDWa或者SDWe，否则无法编译生成正确的下载文件夹。产品系列也可以在“工具-〉项目属性”中设置。
3. SDWn系列编译生成的下载文件夹为“VT_SET_SDWn”，
   SDWa系列编译生成的下载文件夹为“VT_SET_SDWa”，
   SDWe系列编译生成的下载文件夹为“VT_SET”。
   ***必须正确拷贝对应下载文件夹，否则无法下载***

## 1.3 认识组态

VGUS 组态串口屏采用变量驱动，所有显示的数字、字符、图标等都定义为一个变量，并通过上位机软件分配变量存储地址、定义显示格式（如字体、字号大小、颜色、显示位置等），然后生成配置文件并下载保存到串口屏里。需要刷新显示某个变量时，用户单片机仅需将变量内容和变量存储地址通过串口对应发送给串口屏，串口屏会自动按照定义好的显示格式显示。

和过去指令集串口屏相比，VGUS 组态串口屏将显示格式与变量内容分开，在刷新变量时用户单片机只需要发送变量内容即可，而传统的指令集串口屏是要把变量内容和显示格式一起发送的。

VGUS 组态串口屏大大简化了指令数量和格式，仅仅需要 4 条指令 0x80-0x83 就可以完成所有控制和变量显示功能了。

**举例：**在1号界面上显示浮点型变量 “25.3”，VGUS组态串口屏实现步骤：

- **Step1显示配置：**首先通过VGUS组态软件在1号界面上添加一个“数据变量”，设置“数据变量显示格式”的基本属性(包括变量存储地址、显示颜色、大小，显示方式等)，如右所示。然后生成配置文件，并下载到串口屏里。
- **Step2 运行并刷新显示：**需要刷新显示时，只需要通过82指令将变量内容连同变量存储地址一起发送给串口屏即可。

82指令数据格式：　A5 5A 05 82 00 00 00 FD

A5 5A：　串口数据帧帧头（帧头内容用户可以自定义）
05：　数据长度
00 00：　用户定义的25.3这个变量的变量存储地址
00 FD：　25.3这个变量　（小数点是定点显示，即数据“253”的16进制）

| 区域范围设置 | |
|---|---|
| X坐标 | 115 |
| Y坐标 | 179 |
| 宽度 | 144 |
| 高度 | 55 |
| **变量属性** | |
| 名称定义 | 数据变量显示 |
| 描述指针(0x) | FFFF |
| 变量存储地址(0x) | 0000 |
| 初始值 | 0 |
| 变量类型 | int ( 2Byte ) |
| 整数位数 | 2 |
| 小数位数 | 1 |
| 文本颜色 | 255; 0; 0 |
| 字库位置 | 0 |
| X方向点阵数 | 12 |
| 对齐方式 | 居中 |

**说明：**如果切换到2号界面后，再回到1号界面，串口屏会自动从变量存储器里读取变量内容并显示。

图1-3-1 组态屏

VGUS 开发工具是一款组态型、用户图形界面设计软件，VGUS 屏系统架构如图 1-3-2 所示。

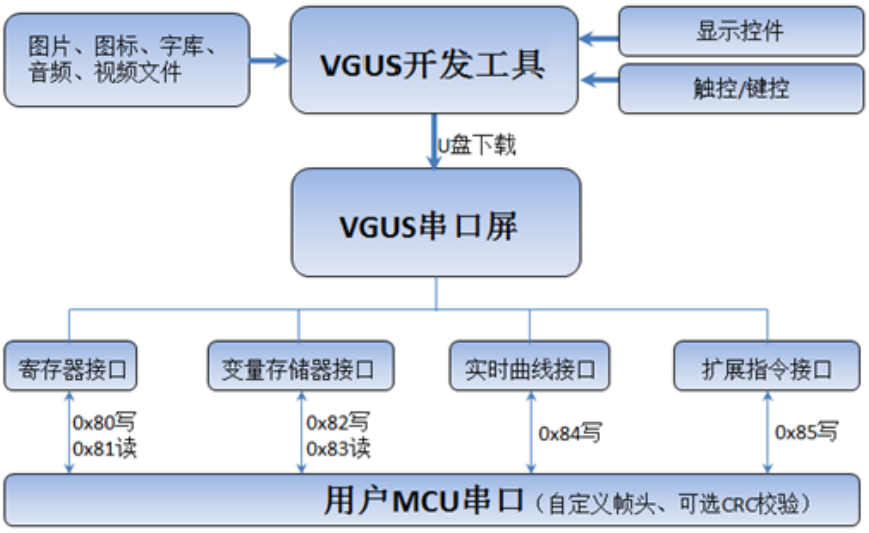



图 1-3-2 VGUS 屏系统架构

所有用到的图片、图标、字库、音频文件需要导入到 VGUS 开发工具中，然后在开发工具中设计显示控件、触摸按钮等，设计完毕后生成下载文件夹并通过 U 盘下载到串口屏里。

VGUS 串口屏与用户 MCU 之前通过串口连接，通过 0x80 和 0x81 指令实现控制、设置等相关功能，通过 0x82、0x83 指令实现显示和触摸等相关功能。指令详细介绍见第 2 章内容。

VGUS 屏开发只需要三个步骤，便可完成复杂的人机交互设计，如图 1-3-3。详细信息见第 7.3 节。

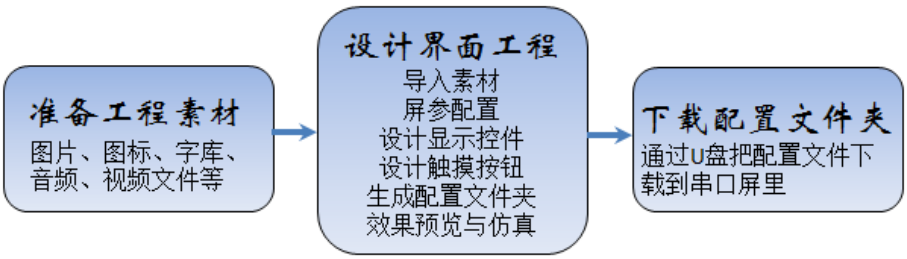



图 1-3-3 VGUS 屏开发步骤

## 1.4 存储空间

为了顺利学习 VGUS 组态串口屏的使用方法，本节内容必须充分理解。

VGUS 屏内部存储空间分为三个部分：寄存器区、变量存储器区和 Flash 存储器。

寄存器区共 256 个单元（地址 0x00~0xFF），每个单元对应一个字节，共 256 字节。用于硬件配置和控制操作，典型功能如背光亮度调节、版本信息读取、发串口指令控制图片显示、音频播放控制、视频播放控制等。用户单片机可以分别通过 0x80、0x81 指令写入和读取寄存器单元内容。寄存器详细介绍见第三章内容。

变量存储器区共 64K 单元（地址 0x0000~0xFFFF），每个单元对应两个字节，共 128K 字节（SDWn 系列共 1K 单元，地址为 0x0000~0x03FF，共 2K 字节）。顾名思义，变量存储器就是用来存储变量值的。用户单片机分别通过 0x82、0x83 指令写入和读取变量存储器单元内容。变量存储器详细介绍见第四章、第五章内容。

Flash 存储器 8M-128M 字节, SDWe 系列 128M 字节，SDWa、SDWn 系列 8M/16M 字节。Flash 存储器主要用于存储配置文件（如变量显示格式）、图片文件、图标文件、字库文件、音频文件、视频文件等。在界面工程设计完毕后，以上内容通过 U 盘/TF 卡下载到串口屏里并永久保存。VGUS 屏采用了 JPG 数据压缩算法和连续存储模式，极大地提高了存储效率和存储图片的数量，对于一般图片应用场合已经足够，例如分辨率为 1024*768 的满幅面图片，128M 字节大约可以存储近千张（实际存储张数与图片色彩有关）。

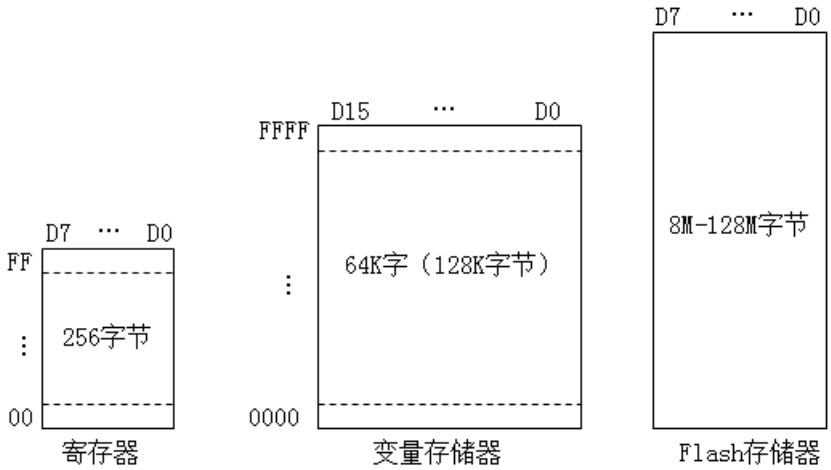



图 1-4-1 SDWa/SDWe 系列存储空间

表 1-4-1 寄存器与变量存储器的区别

| | 寄存器 | 变量存储器 |
|---|---|---|
| 容量 | 256 字节 | 64K 字（128K 字节） |
| 地址范围 | 00-FF | 0000-FFFF |
| 单元大小 | 一个单元对应一个字节 | 一个单元对应两个字节 |
| 读写指令 | 写 0x80/读 0x81 | 写 0x82/读 0x83 |
| 功能 | 单元功能已经定义好，用于硬件配置和控制操作，典型功能如背光亮度调节、版本信息读取、发串口指令控制图片切换、音频播放控制、视频播放控制等。 | 单元功能由用户定义，用于存储变量值。在使用 VGUS 开发工具设计变量时，用户要指定每个变量对应的存储地址(定义变量存储器单元功能)。不同的变量分配的存储地址不能冲突。 |

# 2 串口指令集

VGUS 屏采用变量驱动模式工作，屏的工作模式和GUI状态完全由变量和寄存器来控制。相应的，串口指令也只需要对变量和寄存器进行读、写即可，一共6条指令。

VGUS屏采用异步、全双工串口（UART），每字节数据传送采用10个位：1位起始位、8位数据位、1位停止位，无校验位。串口波特率可通过VGUS 开发工具中“屏参配置”设置。

## 2.1 格式约定

➢ **数据格式**

VGUS屏支持整数（双字节）、无符号整数（双字节）、长整数（4字节）、超长整数（8字节）等数据格式，其数值范围如表2-1所示。

表2-1 数值范围

| 数据格式 | 最小值 | 最大值 |
|---|---|---|
| 整数（双字节） | -32768（0x8000） | +32767（0x7FFF） |
| 无符号整数（双字节） | 0（0x0000） | 65535（0xFFFF） |
| 长整数（4字节） | -2147483648（0x80000000） | +2147483647（0x7FFFFFFF） |
| 超长整数（8字节） | -9223372036854775808 | 9223372036854775807 |

小数采用定点小数表示，用户自定义小数位数，比如0x4D2（1234），规定小数为2位时，表示12.34。

➢ **色彩定义**

VGUS 屏所有颜色数据均为 16 位，两个字节，如表 1-1 所示，其格式为 Red5-Green6-Blue5，即红色占高 5 位，绿色占中间 6 位，蓝色占低 5 位。可以显示的颜色为 $2^{16}$ 色，即 65536 色。

表 2-2 色彩定义

| D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R4 | R3 | R2 | R1 | R0 | G5 | G4 | G3 | G2 | G1 | G0 | B4 | B3 | B2 | B1 | B0 |
| 红色 | | | | | 绿色 | | | | | | 蓝色 | | | | |

例：红色：0xF800，绿色：0x07E0，蓝色：0x001F，白色：0xFFFF，黑色：0x0000

➢ **指令帧结构**

所有的串口指令或者数据采用16进制（HEX）格式。对于双字节数据采用高字节先传送（MSB）的方式。例如0x1234发送时先发送0x12，再发送0x34。

表2-3 指令帧结构

| 指令帧头 | 指令长度 | 指令 | 起始地址 | [数据长度] | [数据内容] | [CRC 校验码] |
|---|---|---|---|---|---|---|

每条指令都包括有指令帧头、指令长度、指令、起始地址四个部分。[数据长度]、 [数据内容]、 [CRC校验码]这三个参数不是每条指令都有。

指令帧头为双字节，其内容用户可以自定义（寄存器R3、RA），通过VGUS开发工具中“屏参配置”设置。用户可以通过设定不同的帧头，实现多台VGUS屏的并联应用。

指令长度为单字节，包括指令及后面所有的内容。

指令为单字节，包括两条寄存器读写指令(0x80、0x81)、两条变量存储器读写指令（0x82、0x83）、一条写曲线缓冲区指令0x84、一条扩展功能指令0x85。

## 2.2 寄存器读写指令 0x80、0x81

VGUS屏有256Byte的寄存器，主要用于相关硬件控制操作，按照字节（Byte）寻址。

起始地址对于寄存器读写指令(0x80、0x81)是单字节，地址范围为0x00~0xff。

数据长度(N)以字节为单位 ，表示连续读寄存器单元个数。

支持顺序读写多个寄存器。

表2-4 串口发送0x80指令帧结构（写寄存器）

| 定义 | 指令帧头 | 指令长度 | 指令 | 起始地址 | 数据内容 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 1(0x80) | 1 | N |

**举例**：寄存器0x03和0x04单元连续写入0x00 、0x01

发送：0xA5 0x5A 0x04 0x80 0x03 0x00 0x01

表2-5 串口发送0x81指令帧结构（读寄存器发送）

| 定义 | 指令帧头 | 指令长度 | 指令 | 起始地址 | 数据长度 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 1(0x81) | 1 | 1(N) |

表2-6 串口接收0x81指令帧结构（读寄存器接收，串口屏发送的）

| 定义 | 指令帧头 | 指令长度 | 指令 | 起始地址 | 数据长度 | 数据内容 |
|---|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 1(0x81) | 1 | 1(N) | N |

**举例**：连续读取寄存器寄存器0x03和0x04单元

发送：0xA5 0x5A 0x03 0x81 0x03 0x02

返回：0xA5 0x5A 0x05 0x81 0x03 0x02 0x00 0x01

表2-7 寄存器读写指令详解

| 功能 | 指令 | 数据 | 说明 |
|---|---|---|---|
| 访问寄存器接口 | 0x80 | 下发：寄存器地址（0x00-0xFF）+写入数据 | 指定地址写寄存器数据 |
| | 0x81 | 下发：寄存器地址（0x00-0xFF）+读取字节长度（0x00-0xFF） | 指定地址开始读指定字节长度的寄存器数据 |
| | | 应答：寄存器地址（0x00-0xFF）+字节数据长度+读取的寄存器数据 | 读寄存器的 VGUS 屏应答 |
| | VGUS 屏有 256Byte 的寄存器，主要用于相关硬件控制操作，按照字节（Byte）寻址。 | | |

## 2.3 变量存储器读写指令 0x82、0x83

VGUS屏有64K word（128K Byte）的变量存储器，按照字（word）寻址操作，地址为0x0000-0xFFFF。用户在规划变量时，要根据变量长度手工分配变量存储器地址。

起始地址对于变量存储器读写指令（0x82、0x83）是双字节，地址范围为0x0000~0xffff。

数据长度（N）以字（双字节）为单位，表示连续读变量存储器单元个数。

数据内容是2N字节，因为每个变量存储单元包含2个字节。

支持顺序读写多个变量存储器单元。

表2-8 串口发送0x82指令帧结构（写变量存储器单元）

| 定义 | 指令帧头 | 指令长度 | 指令 | 起始地址 | 数据内容 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 1(0x82) | 2 | 2N |

**举例**：变量存储器0x0003单元写入0x00 、0x01

发送：0xA5 0x5A 0x05 0x82 0x00 0x03 0x00 0x01

表2-9 串口发送0x83指令帧结构（读变量存储器发送）

| 定义 | 指令帧头 | 指令长度 | 指令 | 起始地址 | 数据长度 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 1(0x83) | 2 | 1(N) |

表2-10 串口接收0x83指令帧结构（读变量存储器接收，串口屏发送的）

| 定义 | 指令帧头 | 指令长度 | 指令 | 起始地址 | 数据长度 | 数据内容 |
|---|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 1(0x83) | 2 | 1(N) | 2N |

**举例**：连续读取变量存储器0x0003和0x0004单元

发送：0xA5 0x5A 0x04 0x83 0x00 0x03 0x02

返回：0xA5 0x5A 0x08 0x83 0x00 0x03 0x02 0x00 0x01 0xff 0xff

表2-11 变量存储器读写指令详解

| 功能 | 指令 | 数据 | 说明 |
|---|---|---|---|
| 访问变量存储器接口 | 0x82 | 下发：变量地址（0x0000-0xFFFF）+写入的变量数据 | 指定变量地址开始写入数据（字数据）到变量存储区 |
| | 0x83 | 下发：变量地址（0x0000-0xFFFF）+读取变量数据字长度（0x00-0x7F） | 从变量存储区指定地址开始读入指定字长度的数据 |
| | | 应答：变量存储器地址+变量数据长度+读取的变量数据 | 读数据存储器的 VGUS 应答 |
| | VGUS 屏采用变量驱动方式，将变量数值和变量显示格式分开。变量显示格式是预先通过配置文件形式下载在显示终端里。而变量数值是通过串口实时传送给显示终端的，变量存储器就是用来存储接收到的变量数值。<br>VGUS 屏有 64K word（128K Byte）的变量存储器，按照字（word）寻址操作，地址为 0x0000-0xFFFF。用户在规划变量时，要根据变量长度手工分配变量存储器地址。 | | |

## 2.4 扩展指令 0x85

表 2-12 直接写显存 0x85 00

| 定义 | 指令帧头 | 指令长度 | 指令 | 数据 | 说明 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 2 (85 00) | X(2byte)+Y(2byte) +DATA0(2byte)+… +DATAn(2byte) | X 为起始位置的 X 坐标，Y 为起始位置的 Y 坐标；<br>DATA 为像素点颜色值，双字节。写入数据超过一行时自动换行。 |

表2-13 串口下载指定文件0x85 01

| 定义 | 指令帧头 | 指令长度 | 指令 | 数据 | 说明 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 2 (85 01) | 5a a5 + filesize(4bytes)+ filename(ASCII) | filesize文件大小，单位字节；<br>filename下载文件的文件名，完整的文件名，含扩展名，<br>如文件名为“1.jpg”，发送的ASCII码应为31 2e 6a 70 67。 |
| | - | - | - | Please Tx file！ | 串口屏正确收到 85 01 指令后，返回提示信息（无帧头） |
| | - | - | - | file | 单片机收到返回提示后，可以开始发送文件（无帧头） |
| | - | - | - | One file Saved OK ！ | 串口屏完整收到文件并保存完成后，返回提示信息（无帧头） |

0x85 01 5a a5 指令下载指定文件，可以通过串口下载各种文件到串口屏里，包括图片、字库、语音、视频等各种文件。如果客户主板可以连接网络，可以用该功能实现远程更新部分显示内容。串口屏固件不能通过该指令实现升级。连接 USB 线时用串口下载文件会导致文件出错。

注意：SDWa/SDWn 系列串口屏不支持该指令功能。

表 2-14 连续播放音频文件 0x85 03

| 定义 | 指令帧头 | 指令长度 | 指令 | 数据 | 说明 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 2 (85 03) | Mode(1byte)+ NUM0(2byte)+… +NUMn(2byte) | Mode定义播放模式：0循环播放，1顺序播放，其他停止播放；<br>NUM指定音频文件名编号（双字节，取值范围0x0000-0xFFFE），<br>如果设为不存在的音频文件名将直接跳过。 |

为提高播放效果，指令中的音频文件格式必须一致，如同为 WAV 或者 MP3 格式。WAV 格式必须用 22KHz、双声道数据，指令中音频文件总大小不得大于 2M。 MP3 格式音频文件总大小没有限制。

0x85 03 指令连续播放指定的多个语音文件。该指令可用于实现窗口/柜台的数字播报（音频文件必须用 WAV 格式），也可以用于娱乐应用（音频文件必须用 MP3 格式）。

## 2.5 CRC 校验

用户可以自定义是否启用CRC检验，通过VGUS开发工具中“屏参配置”设置。启用CRC校验后，所有指令尾部将增加一个双字节的CRC校验码。

CRC校验不包括指令帧头和指令长度，采用ANSI-16（X16+X15+X2+1）格式。

**举例：**带CRC校验方式将寄存器0x03和0x04单元连续写入0x00 、0x01

**发送：**0xA5 0x5A 0x06 0x80 0x03 0x00 0x01 0x18 0x24

| 定义 | 帧头 | 指令长度 | 指令 | 起始地址 | 数据内容 | CRC 校验码 |
|---|---|---|---|---|---|---|
| 举例 | 0xA5,0x5A | 0x06 | 0x80 | 0x03 | 0x00，0x01 | 0x18,0x24 |

当同时启用CRC校验和CRC校验结果应答后（通过VGUS开发工具中“屏参配置”设置），VGUS屏会在CRC校验后自动通过串口外发校验结果。CRC检验结果应答帧结构如表2-16所示。

表2-16 CRC校验结果应答帧结构

| 定义 | 帧头 | 数据长度 | 指令 | 数据 | CRC 校验码 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 1 | 1 | 2 |
| 说明 | 用户自定义 | 固定为 0x02 | 接收到的指令 | 0xff 表示 CRC 校验正确<br>0x00 表示 CRC 校验错误 | |
| 举例 | 0xA5,0x5A | 0x02 | 0x80 | 0xff | 0x20,0x30 |

### CRC 校验程序

```
uint8_t uchCRCHi;     //CRC 高字节
uint8_t uchCRCLo;     //CRC 低字节
static uint8_t auchCRCHi[] = {    /* CRC  高位字节值表*/
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,0x80, 0x41, 0x00, 0xC1, 0x81, 0x40} ;
static uint8_t auchCRCLo[] = {        /* CRC  低位字节值表*/
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06,0x07, 0xC7, 0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD,
```

```
0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A,
0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC, 0x14, 0xD4,0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3,0xF2, 0x32, 0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4,
0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29,
0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED,0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60,0x61, 0xA1, 0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67,
0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68,
0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E,0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71,0x70, 0xB0, 0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92,
0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B,
0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B,0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42,0x43, 0x83, 0x41, 0x81, 0x80, 0x40} ;

uint16_t CRC16(uint8_t * pMsg, uint16_t Len)
{       uint8_t i;
        uchCRCHi = 0xFF;
        uchCRCLo = 0xFF;
        while (Len--)
        {i = uchCRCHi ^ (*pMsg++);
                uchCRCHi = uchCRCLo ^ auchCRCHi[i];
                uchCRCLo = auchCRCLo[i];
        }
        return (uchCRCHi << 8 | uchCRCLo);
}
```

# 3 寄存器

VGUS 屏内部都设计有 256 字节的寄存器，用于实现硬件相关操作，如背光调整、时钟读写、语音播放等。

## 3.1 寄存器一览表

表 3-1-1 寄存器汇总表(***灰色背景部分为 SDWn 系列不支持的***)

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x00 | Version | R | 1 | VGUS 版本号，BCD 码表示，0x40 表示 V4.0 |
| 0x01 | LED_NOW | R/W | 1 | LED 亮度控制寄存器，0x00-0x40 |
| 0x02 | BZ_TIME | W | 1 | 蜂鸣器鸣叫控制寄存器，单位 10ms |
| 0x03-0x04 | PIC_ID | R/W | 2 | 读：当前显示页面 ID；写：切换到指定页面 |
| 0x05 | TP_Flag | R/W | 1 | 0x5A 表示触摸屏坐标有更新；其他表示触摸屏坐标未更新 |
| 0x06 | TP_Status | R | 1 | 0x01 表示第一次按下；0x03 表示一直按下中；0x02 表示抬起；其他表示无效 |
| 0x07-0x0A | TP_Position | R | 4 | 触摸屏按压坐标位置 X_H:L Y_H:L |
| 0x0B | TPC_Enable | R/W | 1 | 0x00 表示触控不启用；其他表示触控启用（上电默认 0xFF） |
| 0x0C-0x0F | RUN_TIME | R | 4 | 上电后运行时间，BCD 码时分秒，其中小时为两个字节，最大 9999:59:59 |
| 0x10-0x1C | R0-RC | R/W | 13 | 屏参配置寄存器映射，当 1D 寄存器配置时，11-1C 寄存器可改写 R1-RC 的数据 |
| 0x1D | CONFIG_EN | W | 1 | 写入 0x5A，R1-RC 重新设置并保存；写入 0xA5，只重新设置，不保存 |
| 0x1F-0x26 | RTC_COM_ADJ | W | 1 | 0x5A 表示用户串口申请改写 RTC 数据，VGUS 修改 RTC 后清零 |
| | RTC_NOW | R/W | 7 | YY：MM:DD:WW:HH:MM:SS,年:月:日:星期:时:分:秒，格式为 BCD 码 |
| 0x27-0X3F | 保留 | | 16 | 未定义 |
| 0x40-0x49 | En_Lib_OP | R/W | 1 | 0x5A 表示申请用户数据文件加载到变量空间，VGUS 操作完后清零。每个 VGUS 周期执行一次读操作 |
| | Lib_OP_Mode | W | 1 | 固定为 0xA0 |
| | Lib_ID | W | 1 | 指定用户数据文件名 0x00-0xff |
| | Lib_Address | W | 3 | 指定数据操作首（字）地址，0x00:00:00-0x01:FF:FF |
| | VP | W | 2 | 指定变量空间的数据操作（首）地址 0x0000-0xFFFF |
| | OP_Length | W | 2 | 操作的（字）数据长度，0x0001-0xFFFF |
| 0x4A-0x4E | Timer0 | R/W | 2 | 16bit 软件定时器,单位 4ms,自减到零停止 |
| | Timer1 | R/W | 1 | 8bit 软件定时器,单位 4ms,自减到零停止 |
| | Timer2 | R/W | 1 | 8bit 软件定时器,单位 4ms,自减到零停止 |
| | Timer3 | R/W | 1 | 8bit 软件定时器,单位 4ms,自减到零停止 |
| | | | | 设置值和实际运行值之间有+/-4ms 误差，比如设置位 2,实际运行值在 4-12ms 之间 |
| 0x4F | Key_code | W | 1 | 用户键码，用于触发配置文件，0x01-0xFF，0x00 表示无效<br>VGUS 处理键码后会自动清零键码寄存器 |

| | | | | |
|---|---|---|---|---|
| 0x50-0x55 | Music_Set | W | 1 | 0x5b:播放语音<br>0x5c:停止播放 |
| | Music_Num | W | 2 | 音频播放设定值（0-4095） |
| | Vol_Adj_En | W | 1 | 0x5A 表示申请调整播放音乐的音量 |
| | Vol | W | 1 | 音量值，范围 0x00-0x40，上电默认值是 0x40 |
| | Vol_Status | R | 1 | 音频播放状态 0x00：播放停止；0x01：播放中 |
| 0x56-0x5F | En_DBL_OP | R/W | 1 | 0x5A 表示用户申请进行数据库操作，VGUS 操作完后清零<br>每个 VGUS 周期执行一次数据库读或写操作 |
| | OP_Mode | W | 1 | 0x50 表示把变量存储器空间数据写入数据库空间<br>0xA0 表示把数据库空间的数据读入变量存储器空间 |
| | DBL_Address | W | 4 | 数据库首（字）地址 |
| | VP | W | 2 | 变量空间的首（字）地址，0x0000-0xFFFF |
| | OP_Length | W | 2 | 数据库操作的（字）长度，0x0001-0xFFFF |
| 0x60-0x6e | Play_Avi_Set | R/W | 1 | 0x5a:申请播放 avi 视频，VGUS 播放完后清零 |
| | Avi_Type | W | 1 | 0x00: 单曲播放 VGUS 屏内视频（默认模式）<br>0x01: 单曲循环播放 VGUS 屏内视频<br>0x02: 顺序循环播放 VGUS 屏内视频<br>0x03: 单曲播放 U 盘内视频<br>0x04: 单曲循环播放 U 盘内视频<br>0x05: 顺序循环播放 U 盘内视频<br>注：视频文件扩展名必须是*.avi；<br>单曲播放时文件名必须为阿拉伯数字,如“123.avi”;<br>顺序播放时文件名可以为字母+数字，如“wuhan123.avi”。 |
| | Play_Position | W | 4 | 视频窗口左上角坐标位置 （X_H，X_L,Y_H,Y_L）<br>注：(0,0)表示居中显示。 |
| | Play_Avi_Num | W | 2 | 通过视频文件名选择播放视频曲目，最多允许 65536 个视频；<br>仅用于单曲播放，顺序播放时无效。 |
| | Vol_Adj_En | W | 1 | 0x5A 表示申请调整播放视频音量 |
| | Vol | W | 1 | 播放视频音量值，范围 0x00-0x3F，上电默认值是 0x3F。 |
| | Play_Control | W | 1 | 0x5A: 播放/暂停<br>对于单曲播放方式，当播放完当前视频后，系统自动跳回到当前图片界面。 |
| | Play_Stop | W | 1 | 0x5A: 停止<br>执行停止播放视频后，系统自动跳回到当前图片界面,也可以按照按钮跳转。 |
| | Play_Next | W | 1 | 0x5a:播放下一首(VGUS 处理后清零, 单曲播放时结束) |
| | Play-Prev | W | 1 | 0x5A：播放前一首 |
| | Play_Status | R | 1 | 0x00=空闲； 0x01=播放中； 0x02=暂停。 |
| 0x6f-0xE8 | 保留 | | 122 | 未定义 |

| | | | | |
|---|---|---|---|---|
| 0xE9 | Scan_Status | R/W | 1 | 读：0x00 表示触摸屏未处于录入状态　0x01 表示触摸屏录入状态；<br>写：如果触摸屏在录入状态下写 0x00，强制退出录入状态。 |
| 0xEA | TPCal_Triger | W | 1 | 写入 0x5A 启动一次触摸屏校准，校准完成后会被 VGUS 清零。 |
| 0xEB | Trendline _Clear | W | 1 | 0x55 表示清除全部 8 条曲线缓冲区数据；<br>0x56-0x5D 表示分别清除 CH0-CH7 通道的曲线缓冲区数据。<br>曲线缓冲区数据清除后，本寄存器会被 VGUS 清零。 |
| 0xEC-0xED | 保留 | | 2 | 保留 |
| 0xEE-0xEF | Reset_Triger | W | 2 | 写入 0x5AA5 导致 VGUS 屏软件复位一次 |
| 0xF0-0xF2 | Release Time | R | 3 | 固件发布日期 YY:MM:DD，年：月：日，格式为 BCD 码<br>如 18：06：27，则代表发布日期为 2018 年 6 月 27 日 |
| 0xF3-0xF6 | | | 4 | 系统自检寄存器 |
| 0xF7-0xF9 | | | 3 | 系统备份寄存器（sysinitial/config/spi），内部使用 |
| 0xFA-0xFF | | | 6 | 保留 |

## 3.2 常用寄存器

### 3.2.1 读取版本号

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x00 | Version | R | 1 | VGUS 版本号，BCD 码表示，0x43 表示 V4.3 |

读取版本信息，串口下发指令 A5 5A 03 81 00 01

返回A5 5A 04 81 00 01 43

说明：A5 5A：帧头

04：指令的字节长度，81 00 01 43 共4字节（不含帧头）

81：读寄存器指令

00：寄存器地址

01：返回数据的字节长度，43为1字节

43：返回的数据信息

### 3.2.2 修改背光亮度

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x01 | LED_NOW | R/W | 1 | LED 亮度控制寄存器，0x00-0x40 |

例如，关闭背光 A5 5A 03 80 01 00

建议：1、设备对功耗要求较高时，可以通过修改屏的背光亮度达到降低功耗的目的；

2、开机亮度以及屏保功能可以通过屏参配置设置。

### 3.2.3 控制蜂鸣器

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x02 | BZ_TIME | W | 1 | 蜂鸣器鸣叫控制寄存器，单位 10ms |

例如，控制蜂鸣器鸣叫2s，发送指令A5 5A 03 80 02 C8

### 3.2.4 切换图片

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x03-0x04 | PIC_ID | R/W | 2 | 读：当前显示页面 ID；写：切换到指定页面 |

例如，切换到2号界面的指令A5 5A 04 80 03 00 02

### 3.2.5 按键触发

| | | | | |
|---|---|---|---|---|
| 0x4F | Key_code | W | 1 | 用户键码，使用用户主板上的键盘来触发显示页面上的按钮（详见第 5 章），有效键码范围 0x01-0xFF，0x00 表示无效。4F 寄存器可以与串口屏外接键盘、触摸屏同时使用。VGUS 处理键码后会自动清零键码寄存器。 |

例如，画面上有个按钮的键码属性设置为01，当用单片机口发送指令“A5 5A 03 80 4F 01”后，串口屏会响应该按钮的操作，执行该按钮被按下的所有操作。

### 3.2.6 时钟RTC

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x1F-0x26 | RTC_COM_ADJ | W | 1 | 0x5A 表示用户串口申请改写 RTC 数据，VGUS 修改 RTC 后清零 |
| | RTC_NOW | R/W | 7 | YY：MM:DD:WW:HH:MM:SS,年:月:日:星期:时:分:秒，格式为 BCD 码 |

➢ **串口读取时钟RTC**

0x20 寄存器开始保存了当前 RTC值，使用 0x81 指令读取。

读取日历（YY:MM:DD:WW:HH:MM:SS）：A5 5A 03 81 20 07

读取时间（HH:MM:SS）：A5 5A 03 81 24 03

时间显示需要使用文本时钟或者表盘时钟控件，详见本文4.4节。

➢ **串口修改（写/设置）时钟RTC**

用 0x80 指令改写 0x1F寄存器为 0x5A，并给 0x20 开始的寄存器写入需要修订的时间，即改写了RTC。

例如把RTC设置为2015-06-01星期一18:56:00，串口发送以下指令：

A5 5A 0A 80 1F 5A 15 06 01 00 18 56 00 VGUS屏会自动换算星期，改写时间时星期可以写任意值。

### 3.2.7 触摸屏校准

所有 VGUS 屏出厂前均已经校准，用户使用前无需逐个再次校准。VGUS 屏提供了两种启动触摸屏校准方法，分别适用于不同应用场合。

➢ **方法1——产品集成**

用户 MCU 通过串口向 VGUS 屏 0xEA 寄存器写入 0x5A 将启动一次触摸屏校准。通过该方法，可以将触摸屏校准功能作为最终设备的一个选项功能，提供给设备操作人员。该方法适用于用户 MCU 发串口指令使用。

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0xEA | TPCal_Triger | W | 1 | 写入 0x5A 启动一次触摸屏校准，校准完成后会被 VGUS 清零 |

触摸屏校准发送指令A5 5A 03 80 EA 5A

➢ **方法2——批量生产**

在 VGUS 开发工具“屏参配置”中，勾选“U 盘下载配置文件后启动触摸校准”，则在每次下载配置文件完毕后，VGUS 屏将立即自动启动一次触摸屏校准。如果客户希望在生产环节，每块屏都再次校准一次，可以使用该方法，该方法适用于批量生产阶段使用。所有 VGUS 屏出厂都已经进行校准，不建议客户再次校准。

无论使用上述哪种方法，一旦触摸屏校准启动后，需要按照屏幕提示操作，依次点击屏幕“左上角”、“左下角”、“右下角”、“右上角”和“中心点”十字交叉点提示的触摸位置；当校准完成时，显示终端会自动进入触摸测试状态，点击触摸屏可观察触摸位置准确度。

### 3.2.9 语音播放寄存器

VGUS 屏支持 WAV 和 MP3 两种音频文件格式，扩展名分别为*.wav 或*.mp3。WAV 格式占用空间大，语音立即输出无延迟，适合按键伴音、数字播报等较短时间播放场合。WAV 格式文件必须选用 22KHz、双声道数据。MP3 格式占用空间小，语音输出稍有延迟（0.5 秒），适合开机音乐、背景音乐、语音提示等较长时间播放场合。

用户通过VGUS组态软件将语音文件添加到工程中，再通过0x80指令写相关寄存器控制语音播放和进行音量调节。

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x50-0x52 | Music_Set | W | 1 | 0x5b:播放语音<br>0x5c:停止播放 |
| | Music_Num | W | 2 | 音乐播放设定值（0-4095） |
| 0x53-0x54 | Vol_Adj_En | W | 1 | 0x5A 表示申请调整播放音乐的音量 |
| | Vol | W | 1 | 音量值，范围 0x00-0x40，上电默认值是 0x40 |
| 0x55 | Vol_Status | R | 1 | 音频播放状态 0x00：播放停止；0x01：播放中 |

举例，一段语音（比如“欢迎光临武汉中显”）保存为6.wav，要以100%音量播放，串口下发：

A5 5A 07 80 50 5b 00 06 5A 40

要停止语音播放，串口下发：A5 5A 05 80 50 5c 00 06

当正在播放一首语音过程中，如果接受到新的播放语音指令，将立即结束当前语音并开始播放新的语音内容。

此外，扩展指令0x85 03提供连续播放音频功能，详细参见2.4节内容。

### 3.2.10 视频播放寄存器

关于视频播放更详细信息请参考文档《VGUS串口屏视频播放功能详解》。

注意：SDWa系列串口屏不支持视频寄存器，仅支持视频控件功能。

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x60-0x67 | Play_Avi_Set | R/W | 1 | 0x5A:申请设置播放器参数 |
| | Avi_Type | W | 1 | 0x00: 单曲播放 VGUS 屏内视频（默认模式）<br>0x01: 单曲循环播放 VGUS 屏内视频<br>0x02: 顺序循环播放 VGUS 屏内视频<br>0x03: 单曲播放 U 盘内视频<br>0x04: 单曲循环播放 U 盘内视频<br>0x05: 顺序循环播放 U 盘内视频<br>注：视频文件扩展名必须是*.avi；<br>单曲播放时文件名必须为阿拉伯数字,如“123.avi”;<br>顺序播放时文件名可以为字母+数字，如“wuhan123.avi”。 |
| | Play_Position | W | 4 | 视频窗口左上角坐标位置 （XH，XL,YH,YL）<br>注：(0，0，0,0)表示居中显示。 |
| | Play_Avi_Num | W | 2 | 通过视频文件名选择播放视频曲目，最多允许 65536 个视频;<br>仅用于单曲播放，顺序播放时无效。 |
| 0x68-0x69 | Vol_Adj_En | W | 1 | 0x5A:申请调整播放视频音量 |
| | Vol | W | 1 | 播放视频音量值，范围 0x00-0x3F，上电默认值是 0x3F。 |
| 0x6a | Play_Control | W | 1 | 0x5A: 播放/暂停<br>对于单曲播放方式，当播放完当前视频后，系统自动跳回到当前图片界面。 |
| 0x6b | Play_Stop | W | 1 | 0x5A: 停止<br>执行停止播放视频后，系统自动跳回到当前图片界面,也可以按照按钮跳转。 |
| 0x6c | Play_Next | W | 1 | 0x5A：播放下一首 |
| 0x6d | Play-Prev | W | 1 | 0x5A：播放前一首 |
| 0x6e | Play_Status | R | 1 | 0x00=空闲； 0x01=播放中； 0x02=暂停。 |

使用视频寄存器，可以通过 0x80 写寄存器指令控制和触控两种方式，来控制视频播放。

**方式一：用户指令控制方式**

用户通过 0x80 指令写寄存器 0x60-0x6d，实现 avi 视频的播放、暂停、继续、停止等功能。

视频寄存器应用举例，在显示屏（10，10）位置单曲播放 U 盘内 01 号视频：

首先发送播放参数　　A5 5A 0A 80 60 5A 03 00 0A 00 0A 00 01，

然后发送播发视频指令　　A5 5A 03 80 6A 5A

**方式二：触控方式**

寄存器 0x61-0x6e 映射于变量存储器 0xff01-0xff0e，即操作用户变量存储器 0xff01-0xff0e 可实现

相同的功能（主要是用来触控实现播放 avi 视频，无须用户 MCU 指令干预）。在播放器界面上设计制作播放/暂停、停止、下一曲、音量调节等按钮，通过“按钮键值返回”控件修改变量存储器 0xff01-0xff0e（寄存器 0x61-0x6e），实现无需用户 MCU 干预下播放 avi 视频；

小技巧：使用视频寄存器实现开机自动播放视频“0.AVI”

通过 VGUS 开发工具，在 0xff0a 单元位置设计一个数据变量控件，并将初值设置为 0x5a，将开机动画视频文件名设置为 0.avi，串口屏上电后将自动播放开机动画 0.avi。

注意：播放视频功能时显示屏上变量不再刷新、上一节中语音播放不可用。

### 3.2.11 用户数据文件

用户数据文件必须命名为“*.bin”，文件名编号为0-255（文件名不可以使用22和23，系统占用）。

**1 通过 U 盘导入用户数据文件**

用户数据文件“*.bin”通过VGUS软件添加到工程里面。通过U盘下载工程文件时，用户数据文件会一并导入到串口屏里。

**2 通过串口读取用户数据文件**

通过寄存器0x40-0x49，可以将用户数据文件里面指定位置数据读出加载到变量存储器里面，如表3-5所示。如果用户单片机需要使用，可以使用0x83指令再从变量存储器中读取。

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x40-0x49 | En_Lib_OP | R/W | 1 | 0x5A 表示申请用户数据文件加载到变量空间，VGUS 操作完后清零。每个 VGUS 周期执行一次读操作 |
| | Lib_OP_Mode | W | 1 | 固定为 0xA0 |
| | Lib_ID | W | 1 | 指定用户数据文件名编号 0x00-0xff |
| | Lib_Address | W | 3 | 指定数据操作首（字）地址，0x00:00:00-0x01:FF:FF |
| | VP | W | 2 | 指定变量空间的数据操作（首）地址 0x0000-0xFFFF |
| | OP_Length | W | 2 | 操作的（字）数据长度，0x0001-0xFFFF |

例如从“80.bin”用户数据文件的 0x 00 00 00 地址开始读取4KW（0x10 00）数据到变量存储器 0x1000 单元开始的位置，串口下发指令：

A5 5A 0C 80 40 5A A0 50 00 00 00 10 00 10 00

读取用户数据文件大约需要花费十毫秒级时间，用户单片机发送连续指令时需要注意插入延时。

### 3.2.12 数据库读写

用户数据库的文件名为“UserDb.bin”，且具有掉电存储用户数据的功能。SDWe系列串口屏允许数据库文件的大小与用户工程文件有关，总空间为114M字节，工程文件越大、允许数据库文件越小。SDWa系列串口屏数据库大小固定为64K字节。

**1 通过串口读写数据库**

用户单片机使用串口访问数据库，需要通过变量存储器实现。通过寄存器0x56-0x5F，可以将变量存储器指定位置的数据写入保存到FLASH里面，也可以将FLASH里面的指定位置数据读出加载到变量存储器里，如下表所示。

| 寄存器地址 | 定义 | R/W | 字节长度 | 说明 |
|---|---|---|---|---|
| 0x56-0x5F | En_DBL_OP | R/W | 1 | 0x5A 表示用户申请进行数据库操作，VGUS 操作完后清零<br>每个 VGUS 周期执行一次数据库读或写操作 |
| | OP_Mode | W | 1 | 0x50 表示把变量存储器空间数据写入数据库空间<br>0xA0 表示把数据库空间的数据读出加载到变量存储器空间 |
| | DBL_Address | W | 4 | 数据库首（字）地址 |
| | VP | W | 2 | 变量空间的首（字）地址，0x0000-0xFFFF |
| | OP_Length | W | 2 | 数据库操作的（字）长度，0x0001-0xFFFF |

例如把变量存储器地址 0000-0100 的数据写入数据库地址 0000 0000-0000 0100，数据长度为 256W（0x01 00），串口下发指令：A5 5A 0C 80 56 5A 50 00 00 00 00 00 00 01 00

把数据库地址 0000 0000-0000 0100 的数据读出保存到变量存储器地址 0000-0100，数据长度为 256W（0x01 00），串口下发指令：A5 5A 0C 80 56 5A A0 00 00 00 00 00 00 01 00

如果用户单片机需要使用数据库里数据，可以先把数据库读出加载到变量存储器，然后再使用0x83指令从变量存储器中读取，反之亦然。

**2 通过 U 盘导出数据库**

VGUS屏工作状态下插入U盘（Fat32格式），VGUS屏会自动将数据库文件“UserDb.bin”拷贝到U盘根目录下。出厂默认状态下，串口屏里面没有数据库文件“UserDb.bin”，只有通过寄存器0x56-0x5F执行了数据库写入操作后，串口屏里面才会生成数据库文件“UserDb.bin”。

数据库保存在 FLASH 存储器里面，因此其具有掉电保存功能。VGUS 串口屏采用的是 NAND 型 FLASH 存储器，具有容量大的优点，但是也具有两点劣势：一是写入次数较少，理论写入次数为 10 万次；二是只支持块写入，写入过程复杂。因此，不推荐将数据库作为重要数据的存储介质使用，也不推荐频繁修改写入数据库！该类应用建议用户单片机外扩一个 24 系列 SPI EEPROM 存储器或者 25 系列 SPI NOR FLASH 存储器。

此外，读写数据库大约需要花费十毫秒级时间，用户单片机发送连续指令时需要注意插入延时。

# 4 显示控件

## 4.0 显示控件介绍

### 4.0.1 显示控件汇总

显示控件通过“VGUS开发工具-》变量配置工具栏”添加，该工具栏各图标定义如图4-0-1所示。

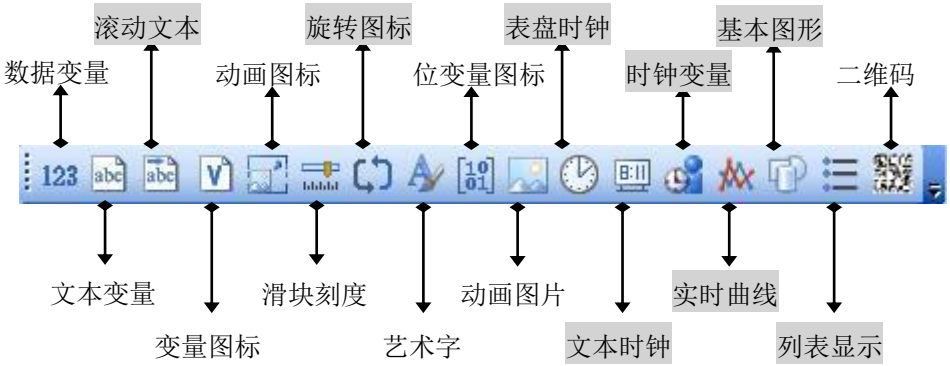



图 4-0-1 变量配置工具栏

表 4-0-1 显示控件汇总表(***灰色背景部分为 SDWn 系列不支持的***)

| 序号 | 功能 | 说明 |
|---|---|---|
| 10 | 数据变量 | 把一个数据变量按照指定格式（整数、小数、是否带单位）用指定字体和大小的阿拉伯数字显示出来。 |
| 11 | 文本变量 | 把字符串按照指定的格式，在指定的文本框显示区域显示。 |
| 14 | 滚动文本 | 从右向左滚动文本显示。 |
| 12_00 | 文本时钟 | 利用串口屏自带的 RTC 时钟外设，按照用户编辑的格式、用文本方式显示公历 RTC。 |
| 12_01 | 表盘时钟 | 利用串口屏自带的 RTC 时钟外设，用表盘方式显示公历 RTC。 |
| 13 | 时钟变量 | 利用用户主板上的 RTC 时钟，以文本方式显示时间。 |
| 00 | 变量图标 | 将一个数据变量的变化范围线性对应一组 ICON 图标显示；当变量变化时，图标也自动相应切换。多用于精细的仪表板、进度条显示。 |
| 01 | 动画图标 | 将一个定值数据变量对应 3 种不同的图标指示状态：不显示、显示固定图标、显示动画图标。 |
| 02 | 滑块刻度 | 将一个数据变量的变化范围对应一个图标（滑块）的显示位置变化。<br>多用于液位、刻度盘、进度表的指示。 |
| 03 | 艺术字变量 | 用 ICON 图标取代字库来显示变量数据。 |
| 04 | 动画图片 | 将一组全屏图片按照指定速度播放。多用于开机界面或屏保。 |
| 05 | 旋转图标 | 把一个数据变量的变化范围线性对应角度数据，然后把一个 ICON 图标按照对应的角度数据旋转后显示出来，多用于指针仪表板显示。 |
| 06 | 位变量图标 | 把一个数据变量的每个位（bit）的 0/1 状态对应 8 种不同显示方案中的两种，<br>用 ICON 图标（或图标动画）来对应显示。<br>多用于开关状态显示，比如风机的运转（动画）、停止（静止图标）。 |

| | | |
|---|---|---|
| 20 | 实时曲线（趋势图） | 结合 0x84 串口写曲线缓冲区数据来自动匹配显示实时曲线（趋势图）。可以指定显示区域、中心轴坐标、显示比例（放大/缩小）可控。 |
| 21 | 基本图形 | 画点、线、矩形、圆形等，详细见 4.3.2 节。 |
| 22 | 列表显示 | 把按照二维数组定义的数据用表格分栏显示出来。 |
| 23 | 二维码显示 | 根据指定内容在屏上显示指定的二维码图形信息。 |
| 24 | 视频控件 | 播放指定视频文件 |

### 4.0.2 显示控件的添加方法

控件设计在VGUS开发工具中完成，所有控件设计都分为2个步骤：第一步：在需要添加相应显示功能的页面下，点击相应控件，把该控件添加到页面上，并拖动控件框调整显示位置和大小。第二步：在该页面上点选已经添加的控件，在右侧窗口中设置控件属性。

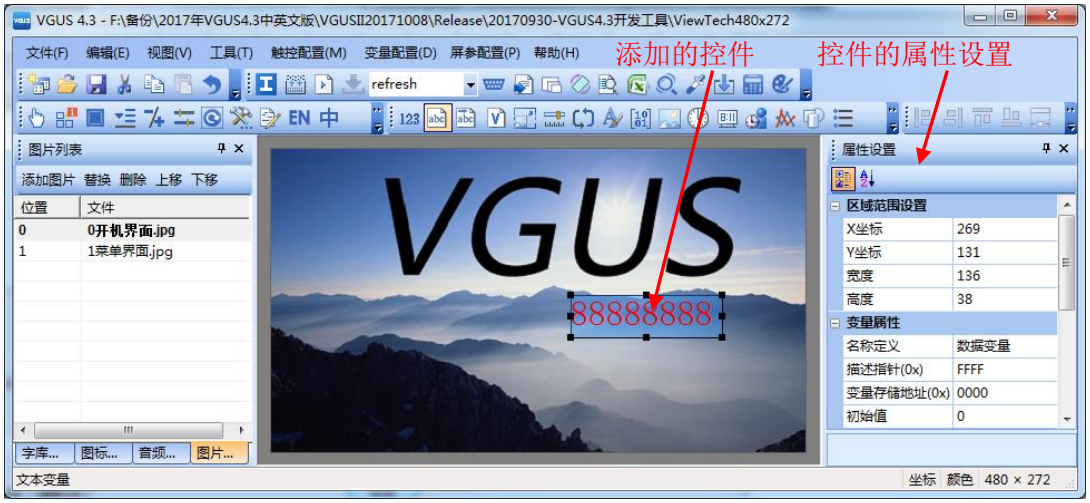



图 4-0-2 点选控件及属性设置举例

### 4.0.2 显示控件的典型属性

显示控件的典型属性如图4-0-3所示，控件属性分为区域范围设置、变量属性和显示格式三大类。区域范围设置与变量属性对所有控件都是类似，而显示格式对不同显示控件可能完全不同。学习显示控件的用法，务必了解清楚变量存储地址的作用和用法、以及显示格式属性栏的功能。鼠标点击指定属性，在属性设置窗口下方会显示关于该属性的详细介绍。

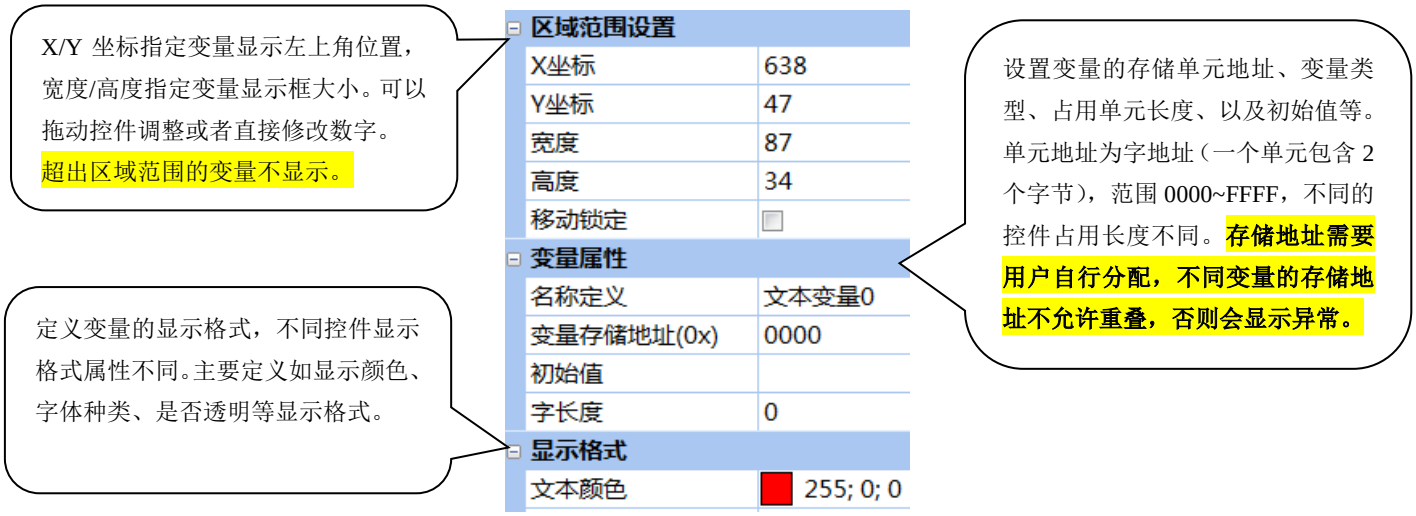



图 4-0-3 显示控件属性说明

## 4.1 数据与文本变量

串口屏预装了0号字库，包含 4*8到64*128 点阵的所有ASCII码字符。当需要用到其它类型字库时，可以在VGUS开发工具中添加常用字库（宋体、微软雅黑和数码管字体）、或者找技术支持索取、或者自行通过TS3软件生成相应字库文件。新字库需要从1开始命名，不要与0号字库重名。

### 4.1.1 数据变量

用于整数、小数的显示。小数显示采用定点方式，先化整并设置好整数位数和小数位数（整数位数和小数位数之和不能超过20），例如显示12.345，需要发送数据12345并设置整数位数为2、小数位数为3。

字地址，范围 0000~FFFF，用于存储变量值，长度由变量类型设定（字地址一个单元包含 2 个字节）。使用 0x82 指令刷新显示内容时需要使用该地址。支持显示负数，负数采用补码、负号要占一位整数位，例如显示“-123”，那么整数位数就要设置为 4 位、发送 0xFF85(-123 的补码)就会显示“-123”。

| 变量属性 | |
|---|---|
| 名称定义 | 数据变量显示 |
| 变量存储地址(0x) | 0020 |
| 初始值 | 0 |
| 变量类型 | int ( 2Byte ) |
| **显示格式** | |
| 整数位数 | 5 |
| 小数位数 | 0 |
| 文本颜色 | 0; 0; 0 |
| 字库位置 | 0 |
| X方向点阵数 | 16 |
| Y方向点阵数 | 32 |
| 对齐方式 | 左对齐 |
| 无效位补零 | ☐ |
| 小数点宽度调整 | ☐ |
| 显示单位 | |
| 描述指针(0x) | FFFF |

设置数据小数位数。VGUS 屏中没有实际意义的小数，实际存储都是整数形式的，小数只是整数中按照指定位置加小数点显示。比如“10.1”，实际存储的、串口通讯的都处理为 0x0065(十进制为 101)，而在显示的时候根据设置的小数位数为 1、显示为“10.1”。。

设置字体点阵大小（Y 方向），必须与上面选择的字库实际点阵一致。

图 4-1-1 数据变量属性设置

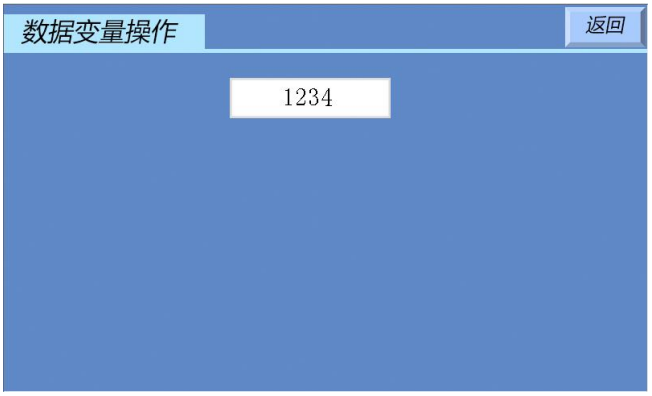



图 4-1-2 数据变量举例

例如，按照图 4-1-1 属性设置，在图 4-1-2 中显示整数“1234”，串口发送指令：A5 5A 05 82 00 20 04 D2

说明：A5 5A：帧头

05：指令字节长度，82 00 20 04 D2 共 5 字节（不含帧头）

82：写变量存储器指令

00 20：变量地址，在控件属性中设置

04 D2：数据内容，1234 的十六进制数据

支持负数显示，负数采用补码、且负号要占一位整数位，例如显示“-123”，那么整数位数就要设置为4位、发送0xFF85(-123的补码)就会显示“-123”。

### 4.1.2 文本变量

用于中英文字符串的显示。对于页面上的固定文字提示信息，建议直接做到背景图片上，一方面在图片上可以设计出更好的字体显示效果，另一方面可以减少页面上的变量，有利于提高串口屏的响应速度。

字地址，范围 0000~FFFF，用于存储变量值，长度由字长度设定（字地址一个单元包含 2 个字节）。使用 0x82 指令刷新显示内容时需要使用该地址。

选择 2 种字库。FONT0 用于显示字符串中的字母和数字，必须为 ASCII 码字库。FONT1 用于显示中文，常用编码方式为 GB2312 或者 GBK。

| 变量属性 | |
|---|---|
| 名称定义 | 文本显示 |
| 变量存储地址(0x) | 0100 |
| 初始值 | |
| 字长度 | 40 |
| **显示格式** | |
| 文本颜色 | 255; 0; 0 |
| FONT0 ID | 0 |
| FONT1 ID | 34 |
| FONT1编码方式 | 0x01=GB2312内码 |
| Y方向点阵数 | 24 |
| 水平间隔 | 0 |
| 垂直间隔 | 0 |
| 字符间距自动调整 | ☑ |
| 描述指针(0x) | FFFF |

1 个汉字占用长度为 1，1 个 ASCII 码字符占用长度为 0.5，范围 1-1024。遇到 0xFFFF 结束符或者显示到文本框尾则不再显示。例如设置为 GB2312 编码，要显示 4 个汉字则字长度至少设置为 4。

指定字库的 Y 方向点阵数，2 种字库的 Y 方向点阵数必须一样，且必须与上面选择的字库实际点阵一致。

图4-1-2 文本变量属性设置

文本变量允许最长文本长度为1024字（2K字节），实际文本显示长度取决于以下三个条件：文本长度限制（属性设置中的字长度）、文本结束符0xFFFF、或者文本框尾，当遇到上述任意一个条件后，显示就会结束。因此，当把字长度设置的大于实际文本显示长度时，在用0x82指令写入文本显示内容，必须要在文本结尾的地方写入文本结束符0xFFFF，否则会在文本尾部显示多余的信息。

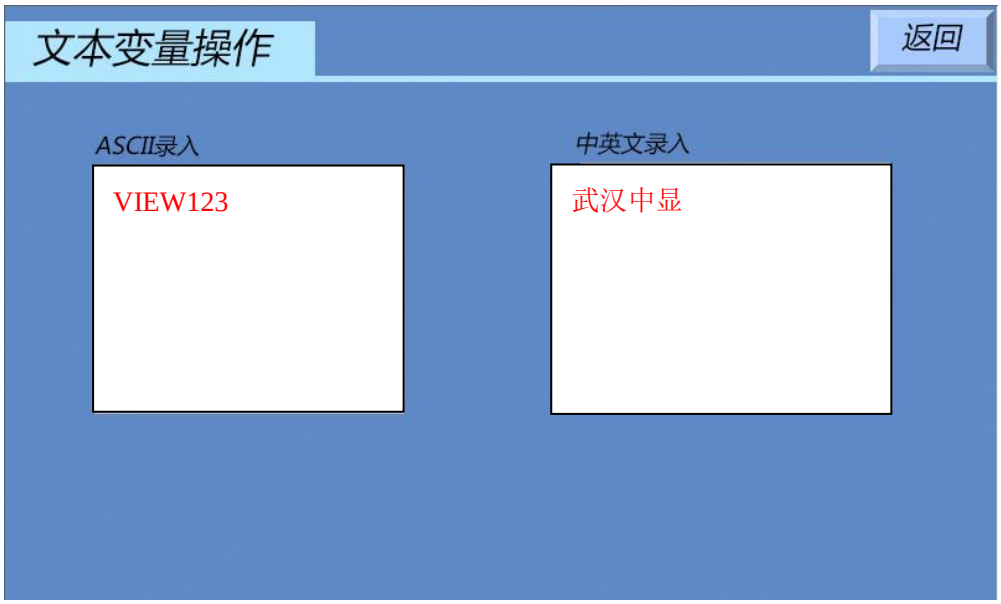



图 4-1-3 文本变量举例

例如，按照图 4-1-2 属性设置，在图 4-1-3 中显示英文加数字“VIEW123”，串口发送指令：

A5 5A 0C 82 01 00 56 49 45 57 31 32 33 FF FF

说明：A5 5A：帧头

0C：指令字节长度，82 01 40 56 49 45 57 31 32 33 FF FF（不含帧头）

82：写变量存储器指令

01 00：变量地址，在控件属性中设置

56 49 45 57 31 32 33：数据内容，“VIEW123”的 ASCII 码

FF FF：文本结束符

例如：显示中文“武汉中显”，串口发送指令：

A5 5A 0D 82 01 40 CE E4 BA BA D6 D0 CF D4 FF FF

说明：A5 5A：帧头

0D：指令字节长度，82 01 40 CE E4 BA BA D6 D0 CF D4 FF FF 共 13 字节（不含帧头）

82：写变量存储器指令

01 40：变量地址，在控件属性中设置

CE E4 BA BA D6 D0 CF D4：数据内容，“武汉中显”的汉字内码

FF FF：文本结束符

如果需要显示日文、韩文、俄文等文字，通常可以使用unicode编码字库。日文也可以使用SJIS编码字库。关于特殊语种显示欢迎联系技术支持。

### 4.1.3 滚动文本

滚动文本的属性设置与文本变量基本类似，只是多了滚屏模式选择和滚屏间距设置。需要注意的是：

1. 变量存储地址开始的三个单元没有使用，文本从“变量存储地址+3”单元开始存储；例如变量存储单元地址为0x0100，我们在用0x82指令写入文本时，要从0x0103单元开始写入。
2. 属性设置中的字长度仅用于VGUS开发工具自动分配变量存储地址和检查单元地址冲突，实际长度以文本结束标志为准，遇到文本结束标志0xFFFF后不再显示。
3. 文本实际长度没有限制，最大允许长度为1024字，使用0x82指令写入文本时必须以 0xFFFF结尾，作为文本结束标记。

字地址，范围 0000~FFFF，用于存储变量值，长度由文本结束标志 0xFFFF 指定（字地址一个单元包含 2 个字节），文本实际长度没有限制。使用 0x82 指令刷新显示内容时需要使用该地址。需要注意上文中的三点。

| 变量属性 | |
|---|---|
| 名称定义 | 滚动文本0 |
| 变量存储地址(0x) | 0000 |
| 初始值 | |
| 字长度 | 0 |
| **显示格式** | |
| 文本颜色 | 255; 0; 0 |
| 对齐方式 | 0x00=左对齐 |
| FONT0 ID | 0 |
| FONT1 ID | 0 |
| FONT1编码方式 | 0x00=8bit 编码 |
| Y方向点阵数 | 16 |
| 字符间隔 | 0 |
| 字符间距自动调整 | ☐ |
| 滚屏模式 | 0x00=从右向左滚 |
| 滚屏间距 | 0 |
| 描述指针(0x) | FFFF |

1 个汉字占用长度为 1，1 个 ASCII 码字符占用长度为 0.5，范围为 1-1024 字。该长度仅用于 VGUS 开发工具自动分配变量存储地址和检查单元地址冲突。实际长度以文本结束标志为准，遇到文本结束标志 0xFFFF 后不再显示，最大允许文本长度为 1024 字。

选择文本滚动方向，仅支持自右向左滚动。

设置每个 VGUS 刷新周期文本滚动的像素点阵数。

图 4-1-4 滚动文本属性设置

## 4.2 图片与图标

VGUS 屏中，约定将分辨率与串口屏分辨率一致的图片称为“图片”，而将所有分辨率小于串口屏分辨率的图片称为“图标”。

在使用图片和图标相关控件以前，必须先设计要图片和图标素材，正确添加到 VGUS 开发工具的文件列表中。关于图片、图标的文件格式要求和管理等信息，请参见本文 7.5 节《文件格式》。

### 4.2.1 变量图标

该控件是把图标编号映射到变量存储单元，通过向变量单元里写入不同的变量，然后串口屏根据变量值检索图标编号，再调用对应图标显示。通常用于串口发 0x82 指令方式，来调用不同的图标在相同的位置显示，例如实现进度条的显示功能、电池电量显示。

需要注意的是：图标编号不是变量值，图标编号与变量值等比均分对应。变量上下限设定的变量范围与图标数量可以不一一对应，串口屏会自动将变量或者图标数进行均分，例如变量为 0 到 100 而图标为 5 到 14，这样变量 0 到 10 对应的图标都是 5。

字地址，范围 0000~FFFF，用于存储变量值，长度为一个单元（字地址一个单元包含 2 个字节）。使用 0x82 指令刷新显示内容时需要使用该地址。

| 变量属性 | |
|---|---|
| 名称定义 | 变量图标 |
| 变量存储地址(0x) | 0120 |
| 初始值 | 0 |
| **显示格式** | |
| 变量下限 | 0 |
| 变量上限 | 9 |
| 图标文件 | 24.ICO |
| 下限对应图标 | 5 |
| 上限对应的图标 | 14 |
| 图标显示模式 | 显示背景 |
| 描述指针(0x) | FFFF |

选择 Icon 文件、指定图标范围。

支持透明显示和带背景显示，透明显示要求素材必须是 BMP 格式。

图4-2-3 变量图标属性设置

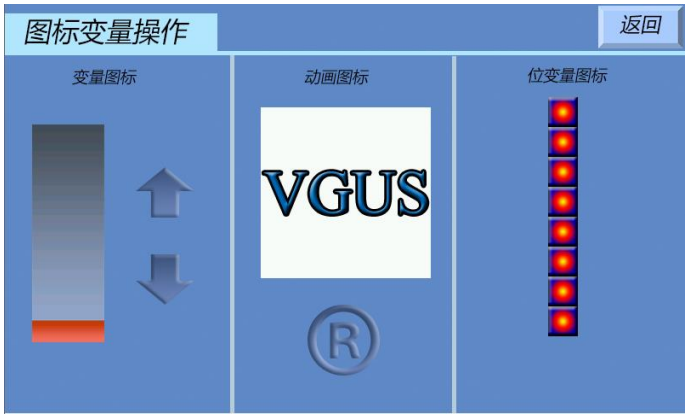



图 4-2-4 变量图标应用举例

例如，按照图 4-2-1 图标素材、图 4-2-3 属性设置，为让上图中左边进度条显示 10 种不同状态，串口可以分别发送指令：

A5 5A 05 82 01 20 00 00

A5 5A 05 82 01 20 00 01

……..

A5 5A 05 82 01 20 00 09

说明：A5 5A：帧头

05：指令字节长度，82 01 20 00 01 共 5 字节

82：写变量存储器指令

01 20：设置的变量地址，如上图变量属性所示

00 00：设置的变量值，变量的上下限 0-9 与编号为 5-14 的图标一一对应，即发送 00 00 显示 5 号图标，

00 01：显示 6 号图标，以此类推。

### 4.2.2 动画图标

该控件是在显示屏某一区域连续循环显示一串图标，实现动画效果，动画播放时间间隔为一个 VGUS 刷新周期（约 100 毫秒）。变量只有 2 个取值：一个为开始值，一个为停止值。当通过 0x82 指令向变量单元写入开始值时，动画开始播放；当写入停止值时，动画停止播放。

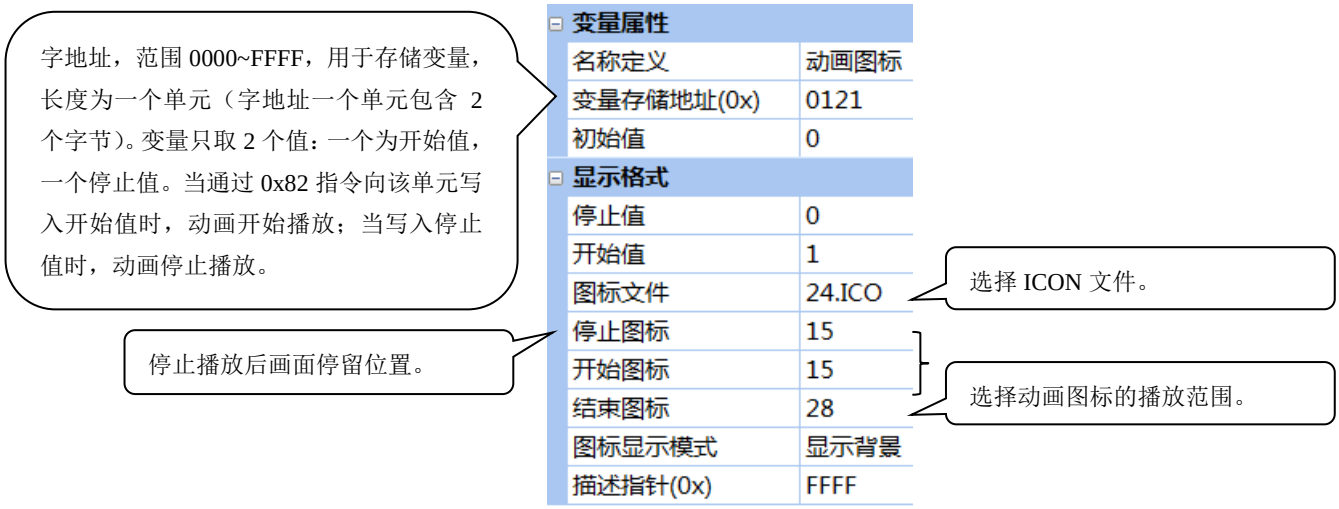



图 4-2-5 动画图标属性设置

按照图 4-2-5 设置：

当向 0121 单元写入 1 后，将把 24.ICO 文件中的 15 到 28 号图标循环播放。

当向 0121 单元写入 0 后，动画停止播放，画面停留到 15 号图标上。

当向 0121 单元写入其它值时，不显示图标和动画。

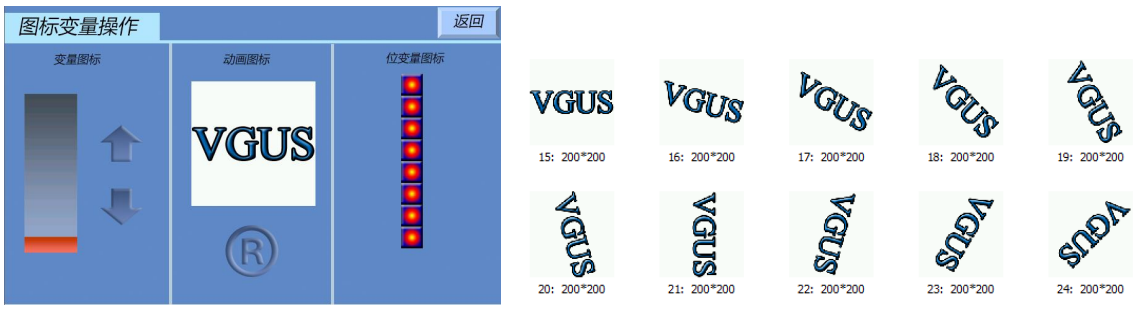



图 4-2-6 动画图标举例

为在上图中间位置显示旋转的“VGUS”动画，需要准备右侧的图标素材。按照图 4-2-5 所示设置，为让上图中间动画图标显示动画效果，可以发送指令：

开始动画指令 A5 5A 05 82 01 21 00 01

停止动画指令 A5 5A 05 82 01 21 00 00

说明：A5 5A：帧头

05：指令字节长度，82 01 21 00 01 共 5 字节

82：写变量存储器指令

01 21：设置的变量地址，如上图变量属性所示

00 00：设置的变量开始值， 发送开始值后循环播放开始图标到结束图标，发送停止值后显示停止图标。

**小技巧：**如果把图 4-2-5 中的初始值设置为 1，就可以实现开机动画，上电自动播放。

### 4.2.3 动画图片

与动画图标不同，动画图片播放整张图片、播放的起始/终止图片和播放时间间隔可以在属性中设置、播放到终止位置图片后自动结束播放。需要特别注意的是，动画图片没有分配变量存储地址单元，显示动画图片控件所在的页面，将自动启动播放。

图 4-2-7 动画图片属性设置

上图设置从第 35 号图片开始，播放到第 49 号图片结束，播放时间间隔为 5*10mS=0.05 秒。

启动播放方法：显示动画图片控件所在的页面，将自动启动播放。

停止播放方法：播放到终止图片位置自动停止播放。

循环播放方法：在终止图片页面上也添加一个属性一样的动画图片控件。

串口发送指令切换图片或者触控按钮切换图片可以结束循环播放。

开机自动播放：在 0 号图片上（开机页面上）添加动画图片控件。

注意：动画图片控件所在页面无法驻留显示。

### 4.2.4 滑动刻度

该控件是指定一个图标（滑块），在水平方向或者垂直方向上、在指定范围内变化位置显示（滑动），显示位置通过变量设定。通常可以配合拖动调节控件使用（详见 5.4.2 节）。需要注意的是：变量值不是坐标，滑块显示位置与变量值等比均分对应。起始/终止刻度值设定的变量范围与滑动区域坐标可以不一一对应，串口屏会自动将变量或者滑动区域坐标进行均分后对应。

字地址，范围 0000~FFFF，用于存储变量，长度为一个单元（字地址一个单元包含 2 个字节）。使用 0x82 指令刷新滑块显示位置时需要使用该地址。

最小变量值。横向滑动图标对应位于最左边，纵向对应位于最上边。

选择滑动方向：横向或者纵向。

| 变量属性 | |
|---|---|
| 名称定义 | 滑块刻度指示 |
| 变量存储地址(0x) | 0180 |
| 变量模式 | 指向一个整型变量 |
| 初始值 | 0 |
| **显示格式** | |
| 起始刻度值 | 0 |
| 终止刻度值 | 1000 |
| 刻度模式 | 横向刻度条 |
| 图标文件 | 24.ICO |
| 滑动图标 | 0 |
| 图标显示模式 | 透明 |
| Y坐标前移偏移量 | 230 |
| X坐标前移偏移量 | 0 |
| 描述指针(0x) | FFFF |

最大变量值。横向滑动图标对应位于最右边，纵向对应位于最下边。

选择滑动图标（滑块）。

图 4-2-8 滑动刻度属性设置

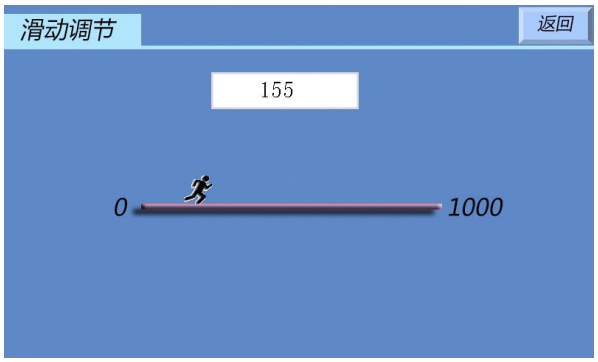



图 4-2-9 滑动刻度举例

为在上图中实现人形滑块显示，可以横线上方添加一个滑块刻度控件，属性设置如图 4-2-8。让滑动刻度显示在指定位置，串口可以发送指令：

A5 5A 05 82 01 80 00 00

A5 5A 05 82 01 80 00 01

……..

A5 5A 05 82 01 80 03 E7

A5 5A 05 82 01 80 03 E8

说明：A5 5A：帧头

05：指令字节长度，82 01 80 00 01 共 5 字节

82：写变量存储器指令

01 80：设置的变量地址，如上图变量属性所示

00 00：设置的变量刻度值，即从 00 00 变化到 03 E8

### 4.2.5 艺术字变量

该控件与数据变量应用一样，都是可以实现一个数据显示。与数据变量不同的是，艺术字变量使用图标（“0123456789-.”共 12 个图标）代替字库，可以实现各种特殊字体效果、特殊颜色效果的数据显示。常见的数码管字体建议使用数据变量实现，显示速度更快。

字地址，范围 0000~FFFF，用于存储变量值，长度由变量类型设定（字地址一个单元包含 2 个字节）。使用 0x82 指令刷新显示内容时需要使用该地址。

| 变量属性 | |
|---|---|
| 名称定义 | 艺术字 |
| 变量存储地址(0x) | 0160 |
| 初始值 | 0 |
| 变量类型 | int（2Byte） |
| **显示格式** | |
| 整数位数 | 5 |
| 小数位数 | 0 |
| 图标文件 | 24.ICO |
| 起始图标 | 29 |
| 图标显示模式 | 透明 |
| 对齐方式 | 右对齐 |
| 描述指针(0x) | FFFF |

VGUS 屏中没有实际意义的小数，实际存储都是整数形式的，小数只是整数中按照指定位置加小数点显示。比如“10.1”，实际存储的、串口通讯的都处理为 0x0065(十进制为 101)，而在显示的时候根据设置的小数位数为 1、显示为“10.1”。

指定数字 0-9、中划线、小数点对应的图标。起始图标对应数字 0，其它顺延。

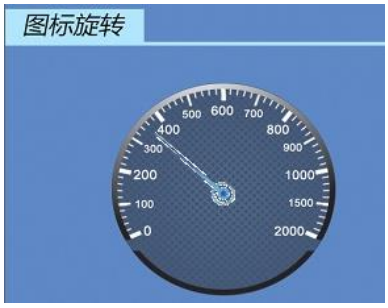
图 4-2-10 艺术字变量属性设置

图 4-2-11 艺术字变量举例

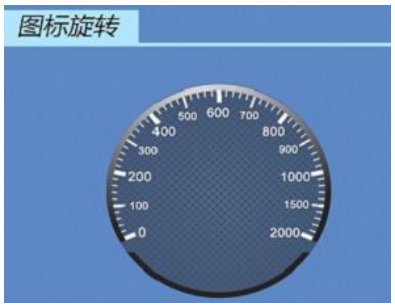
为实现上图中的数据显示效果，只能使用艺术字变量。首先要制作图标素材如上图右侧。在文本框位置添加一个艺术字变量，属性设置参照 5-2-10，当向 0160 单元写入“345”后，就可以显示出上图艺术字出来。

### 4.2.6 旋转图标

该控件控件通常用于仪表盘的显示，需要预先设计一张表盘背景图片和一个指针图标，指针图标根据变量值确定显示角度，指针图标旋转中心坐标可以设置、指针图标旋转中心在表盘背景图片上的显示位置也可以设置。




表盘效果




表盘背景图片

指针图标

图 4-2-12 旋转图标举例

字地址，范围 0000~FFFF，用于存储变量，长度为一个单元（字地址一个单元包含 2 个字节）。使用 0x82 指令刷新指针（图标）显示角度时需要使用该地址。

| 变量属性 | |
|---|---|
| 名称定义 | 图标旋转 |
| 变量存储地址(0x) | 0040 |
| 变量模式 | 指向一个整型变量 |
| 初始值 | 0 |
| **显示格式** | |
| 图标文件 | 24.ICO |
| 图标ID | 1 |
| 图标显示模式 | 透明 |
| 图标旋转中心 | (122,15) |
| 屏幕旋转中心 | (272,264) |
| 起始值 | 0 |
| 终止值 | 2000 |
| 起始角度 | 0 |
| 终止角度 | 400 |
| 描述指针(0x) | FFFF |

选择指针图标。

设置指针旋转中心。

变量最小、最大值。

变量最大值对应的旋转角度。1-720，单位 0.5 度。

变量最小值对应的旋转角度。0-719，单位 0.5 度。

图 4-2-13 旋转图标属性设置

需要注意的是：变量值不是角度值，指针显示角度与变量值等比均分对应。起始/终止值设定的变量范围、与起始/终止角度角度设定的旋转角度可以不一一对应，串口屏会自动将变量或者转动角度均分后对应。

按照上图属性设置，通过向 0040 单元写入不同的值，就可以控制指针的旋转显示位置。旋转始终约定为“顺时针”转动，即 终止旋转角度 必须大于 起始旋转角度(如果终止旋转角度小于起始旋转角度，系统处理时会自动加上 360°)。

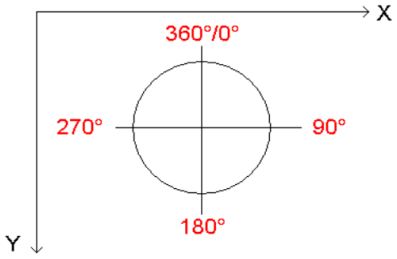
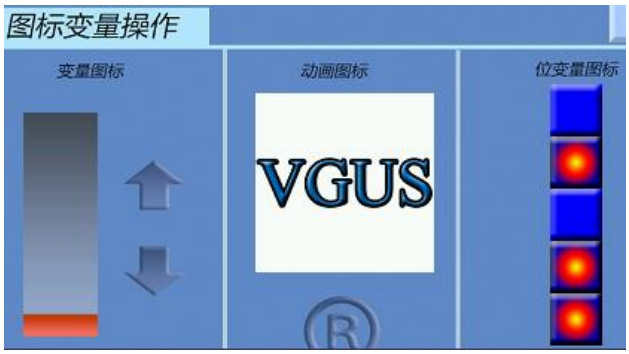



图 4-2-14 旋转方向

### 4.2.7 位变量图标

该控件是把一个数据变量（双字节/共 16 个位）的每个位（bit）的 0/1 状态对应 8 种不同显示方案（显示模式）中的两种，用 ICON 图标（或图标动画）来对应显示。多用于开关状态显示、报警指示灯的显示、风机状态显示（运转为动画、停止为静止图标）。图 4-2-15 中，有 5 个指示灯状的图标，就是使用位变量图标实现的。该例中使用了变量图标中 16 个位的低 5 位，位值为 0 显示的是蓝色方块图标，位值为 1 显示的是带有红色指示灯的图标，5 个图标自上向下依次排列。



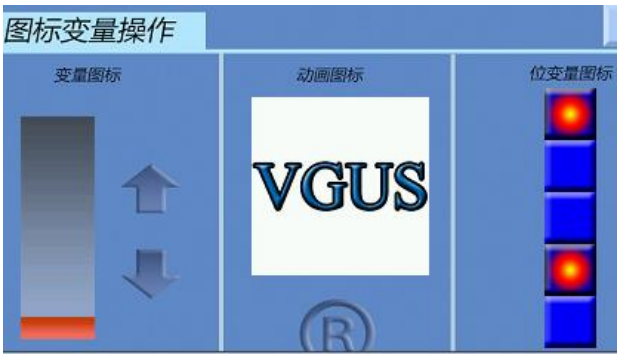




图 4-2-15 位变量图标典型应用举例

表 4-2-1 八种显示方案（显示模式）

| 显示模式 | 位值 | |
|---|---|---|
| | 0 | 1 |
| 0x00 | ICON0S | ICON1S |
| 0x01 | ICON0S | 不显示 |
| 0x02 | ICON0S | ICON1S-ICON1E 动画 |
| 0x03 | 不显示 | ICON1S |
| 0x04 | 不显示 | ICON1S-ICON1E 动画 |
| 0x05 | ICON0S-ICON0E 动画 | ICON1S |
| 0x06 | ICON0S-ICON0E 动画 | 不显示 |
| 0x07 | ICON0S-ICON0E 动画 | ICON1S-ICON1E 动画 |

表 4-2-2 位变量图标排列模式（16 个图标的排列方向）

| 排列模式 | 说　明 |
|---|---|
| 0x00 | 水平方向排列，自左向右、高位在左，不保留未启用位的图标位置； |
| 0x01 | 垂直方向排列，自上向下、高位在上，不保留未启用位的图标位置； |
| 0x02 | 水平方向排列，自左向右、高位在左，保留未启用位的图标位置； |
| 0x03 | 垂直方向排列，自上向下、高位在上，保留未启用位的图标位置。 |

字地址，范围 0000~FFFF，用于存储变量值，长度为一个单元（字地址一个单元包含 2 个字节）。使用 0x82 指令刷新显示内容时需要使用该地址。

字地址，范围 0000~FFFF，用于存储变量值，长度为 2 个单元（字地址一个单元包含 2 个字节）。该地址单元用户禁止访问，用于存储位变量图标显示的辅助数据。

设置位变量图标中 16 个位哪些启用，为 0 相应位不启用，为 1 相应位启用。

图标排列模式，自上向下，不保留未启用的图标位置。排列间隔定义相邻图标的中心距，单位像素。

选择图标，图标用途见表 4-2-1。

| 变量属性 | |
|---|---|
| 名称定义 | 位变量图标 |
| 变量存储地址(0x) | 0122 |
| 辅助地址 | 0000 |
| 初始值 | 0 |
| **显示格式** | |
| 显示位 | 00FF |
| 显示模式 | 0x00 |
| 排列模式 | 0x01 |
| 排列间隔 | 40 |
| 图标文件 | 24.ICO |
| ICON0S | 41 |
| ICON0E | 0 |
| ICON1S | 42 |
| ICON1E | 0 |
| 图标显示模式 | 透明 |
| 描述指针(0x) | FFFF |

图 4-2-16 位变量图标属性设置

图 4-2-15 中，在每个图标位置还定义了一个增量调节按钮，点击后，控制位变量图标切换显示。控件设计如图 4-2-17，在位变量图标的每个图标上还设计了一个增减调节按钮，实现复选功能。增量调节按钮 0 的属性设置如图 4-2-18，变量存储地址与位变量图标的一样，变量模式为位调节。

图 4-2-17 位变量图标控件设计举例

| 变量属性 | |
|---|---|
| 变量存储地址(0x) | 0122 |
| 变量模式 | 位调节 |
| 写入位 | 4 |
| 调节方式 | ++ |
| 调节步长 | 1 |
| 按键模式 | 按键单次调节 |
| 下限 | 0 |
| 上限 | 1 |
| 逾限处理方式 | 循环 |

图 4-2-18 增量调节按钮属性设置

## 4.3 图形变量

### 4.3.1 实时曲线（趋势图）

VGUS 屏有一个 8K Word、可以存储 8 条曲线的缓冲区，用于用户简单、快速显示曲线。曲线缓冲区的数据都是16位无符号数。8K word的曲线缓冲区不占用64K word的变量存储器空间，也没有分配变量存储地址。

◆ **属性设置**

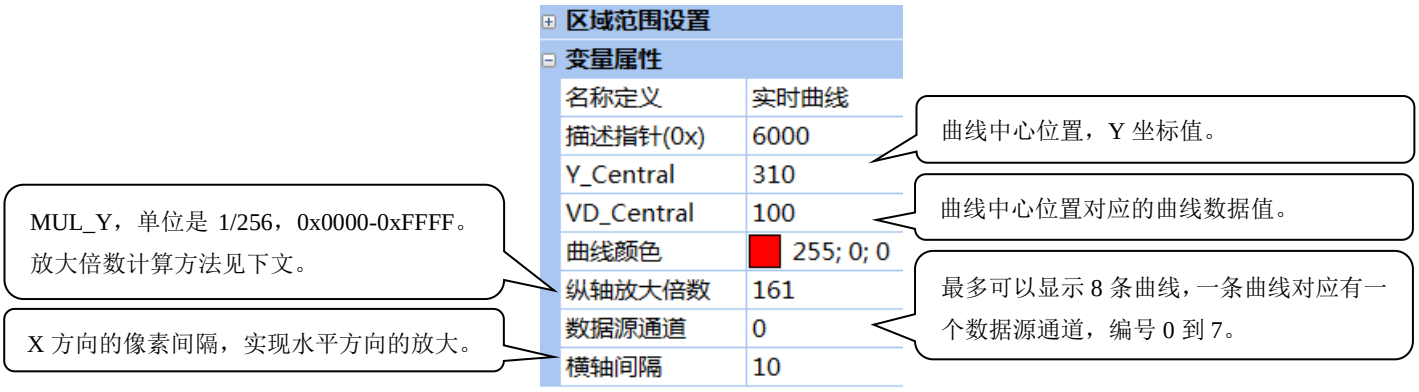



图4-3-1 实时曲线属性设置

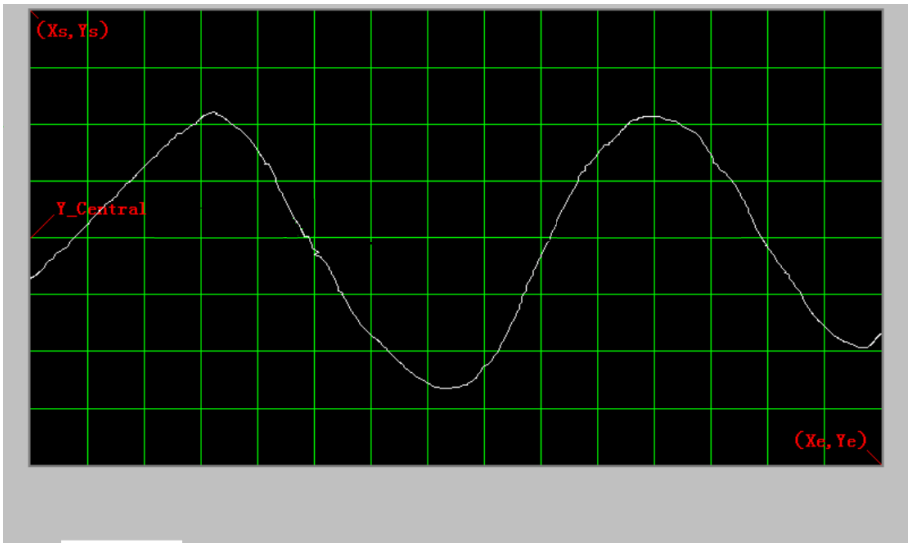



图4-3-2 实时曲线显示举例

首先在属性设置中定义曲线显示区域，如上图网格区域，左上角坐标（Xs,Ys）、右下角坐标（Xe,Ye）。然后设置波形中心线对应在显示屏上的Y坐标、以及中心线对应的曲线数据值，这样就确定了波形显示位置。曲线水平方向放大倍数直接通过调整横轴间隔实现。

满量程曲线纵轴放大倍数计算方法：

MUL_Y=（Ye-Ys）*256/（Vmax-Vmin）

Ye,Ys 为曲线窗口的 Y 坐标，

Vmax,Vmin 为曲线数据的最大,最小值。

例如一个12bit A/D采集数据（Vmax=4095 Vmin=0）要对应在Ys=50,Ye=430的屏幕区域满量程显示，那么：MUL_Y=(430-50)*256/(4095-0)=23.7　向下舍入取 23。

如果需要显示粗线条的曲线，可以把做好的曲线控件复制一个出来，把属性设置中的X坐标加1、

Y_Central加1、其它属性不变，这样就会把曲线复制并平移一个像素，看起来就是曲线变粗了。

利用第6章介绍的描述指针的方法：

- 结合增量调节按钮，把按钮的变量存储地址设置与纵轴放大倍数或者水平间隔存储地址一样，可以实现通过增量调节按钮放大缩小曲线显示。无需用户程序干预。
- 结合拖动调节按钮，把按钮的变量存储地址设置与Y_Central的存储地址一样，可以实现通过拖动调节按钮上下移动曲线。无需用户程序干预。

### ◆ 曲线缓冲区写指令0x84

表4-3-1 曲线缓冲区写指令0x84帧结构

| 定义 | 指令帧头 | 指令长度 | 指令 | 曲线通道 | 曲线数据 |
|---|---|---|---|---|---|
| 长度（字节） | 2 | 1 | 1(0x84) | 1（CH_Mode） | 2n（DATA0（Word）+…+DATAn） |

表4-3-2 曲线通道CH_Mode定义

| 位 | CH_Mode.0 | CH_Mode.1 | CH_Mode.2 | CH_Mode.3 | CH_Mode.4 | CH_Mode.5 | CH_Mode.6 | CH_Mode.7 |
|---|---|---|---|---|---|---|---|---|
| 通道 | 通道 0 | 通道 1 | 通道 2 | 通道 3 | 通道 4 | 通道 5 | 通道 6 | 通道 6 |

- ✧ CH_Mode的每个位（bit）对应 1 个通道；
- ✧ CH_Mode的相应位值为1表示该条曲线存在，为0表示不存在，允许同时存在多条曲线；
- ✧ 曲线数据按照低通道数据在前排列，例如CH_Mode=0x83（10000011B），表示曲线数据格式为：

（通道0+通道1＋通道7）＋…＋（通道 0＋通道1＋通道 7）。

**用户单片机通过0x84指令、按照通道号把曲线数据发送给串口屏。当串口屏收到0x84指令后，接收到的曲线数据总是靠曲线窗口右侧显示、之前的曲线会向左移动、超出窗口长度部分的曲线会移出。**

### 4.3.2 基本图形（绘图）

表 4-3-3 基本图形指令汇总

| 图形指令 | 功能 | 说明 |
|---|---|---|
| 0x0001 | 置点 | 置点（x,y,color） |
| 0x0002 | 端点连线 | 端点连线（color,(x0,y0),…(xn,yn)） |
| 0x0003 | 矩形 | 显示矩形，颜色和位置、大小可控 |
| 0x0004 | 矩形填充 | 填充指定的矩形区域，填充颜色和位置、大小可控 |
| 0x0005 | 画圆 | 显示整圆弧，颜色和位置、大小可控 |
| 0x0006 | 图片剪切粘贴 | 从指定图片上剪切一个区域粘贴到当前显示页面上 |
| 0x0007 | ICON 图标显示 | ICON 图标显示，图标库可以选择 |
| 0x0008 | 封闭区域填充 | 封闭区域填充，种子点坐标、填充颜色可控 |
| 0x0009 | 频谱显示 | 根据变量数据显示频谱（垂直线条），线条颜色、位置可控 |
| 0x000A | 线段显示 | 根据变量数据连接线段，端点、颜色可控 |
| 0x000B | 圆弧显示 | 显示圆弧，半径、颜色、起止角度可控 |
| 0x000C | 字符显示 | 根据变量数据进行单个字符显示 |
| 0x000D | 矩形区域 XOR | 对指定的矩形域位图数据用指定颜色进行 XOR 操作，多用于高亮显示 |
| 0x000E | 双色位图显示 | 变量存储器数据看成双色位图数据，0/1 对应颜色可指定，多用于自定义光标 |
| 0x000F | 位图显示 | 变量存储器数据位 65K 色位图数据，多用于实时图标（照片）下载显示。 |
| 0x0010 | 区域放大粘贴 | 把指定区域放大 1 倍粘贴到指定位置，多用于配合 0F 指令实现照片实时显示 |

图 4-3-3 基本图形属性设置

基本图形显示的属性设置非常简单，如图 4-3-3 所示。区域范围设置必须足够大，超出范围的图形不显示。变量属性中主要设置的也就是变量存储地址，需要显示的图形数据内容需要写入该地址单元中，实际占用地址单元数量取决于图形数据内容的多少，不同基本图形指令（表 4-3-3）的图形数据内容格式不同，参见表 4-3-4、表 4-3-5。

表4-3-4 图形数据内容格式定义

| 地址（字） | 定义 | 说明 |
|---|---|---|
| VP | CMD | 双字节，图形指令 |
| VP+1 | Data_Pack_Num_Max | 双字节，最大数据包数目，连线指令（0x0002） 定义为连线线条数目，即顶点数－1 |
| VP+2 | DATA_Pack | 双字节，数据包，参见表4-3-5 |

基本图形指令举例：a5 5a 15 82 20 00 00 02 00 02 f8 00 02 3B 00 A4 02 EE 01 18 02 C2 01 B9

a5 5a ：指令帧头

15 ：指令长度，21 个字节，单位字节

82 ：写变量存储器指令

20 00 ：变量地址

00 02 ：端点连线的图形指令，　CMD

00 02 ：连线线条数目，　Data_Pack_Num_Max

f8 00 ：连线的颜色 红色，双字节

02 3B 00 A4 ：连线第一个点的坐标

02 EE 01 18 ：连线第二个点的坐标

02 C2 01 B9 ：连线第三个点的坐标

DATA_Pack/数据包

图形数据内容

表 4-3-5 图形指令数据包格式定义

| 图形指令 | 绘图功能 | 绘图数据包格式说明（偏移量和长度单位均为字（word）） | | | |
|---|---|---|---|---|---|
| | | 偏移量 | 长度 | 定义 | 说明 |
| 0x0001 | 置点 | 0x00 | 2 | (x,y) | 置点坐标位置，x坐标高字节为判断条件 |
| | | 0x02 | 1 | color | 置点颜色 |
| 0x0002 | 端点连线 | 0x00 | 1 | color | 线条颜色 |
| | | 0x01 | 2 | (x,y) 0 | 连线顶点0坐标，X坐标高字节为判断条件 |
| | | 0x03 | 2 | (x,y) 1 | 连线顶点1坐标，X坐标高字节为判断条件 |
| | | 0x01+2*n | 2 | (x,y) n | 连线顶点n坐标，X坐标高字节为判断条件 |
| 0x0003 | 矩形 | 0x00 | 2 | (x,y) s | 矩形框左上角坐标，x坐标高字节为判断条件 |
| | | 0x02 | 2 | (x,y) e | 矩形框右下角坐标 |
| | | 0x04 | 1 | color | 矩形颜色 |
| 0x0004 | 矩形填充 | 0x00 | 2 | (x,y) s | 矩形域左上角坐标，X坐标高字节为判断条件 |
| | | 0x02 | 2 | (x,y) e | 矩形域右下角坐标 |
| | | 0x04 | 1 | color | 矩形域填充颜色 |
| 0x0005 | 整圆弧显示 | 0x00 | 2 | (x,y) | 圆心坐标，X坐标高字节为判断条件 |
| | | 0x02 | 1 | Rad | 半径 |
| | | 0x03 | 1 | color | 圆颜色 |
| 0x0006 | 图片区域剪切、粘贴 | 0x00 | 2 | Pic_ID | 剪切图片区域所在页面ID，高字节为判断条件 |
| | | 0x02 | 2 | (x,y) s | 剪切图片区域左上角 |
| | | 0x04 | 2 | (x,y) e | 剪切图片区域右下角 |
| | | 0x06 | 2 | (x,y) | 剪切图片区域粘贴到当前页面的坐标位置，左上角坐标 |
| 0x0007 | ICON 图标显示 | 0x00 | 2 | (x,y) | 显示坐标位置，x坐标高字节为判断条件 |
| | | 0x02 | 1 | ICON_ID | 图标ID，图标库位置由指令高字节指定<br>图标固定为不显示背景色 |
| 0x008 | 区域填充 | 0x00 | 2 | (x,y) | 种子点坐标，x坐标高字节为判断条件 |
| | | 0x02 | 1 | color | 填充颜色 |
| 0x0009 | 频谱显示（垂直线条） | 0x00 | 1 | Color0 | 把(X0,Y0s)(X0,Y0e)用 Color0 颜色连线，X0 高字节为判断条件 |
| | | 0x01 | 3 | X0,Y0s,Y0e | |
| 0x000A | 线段显示 | 0x00 | 1 | Color | 把（Xs,Ys）（Xe,Ye）用 Color 颜色连线，Xs 高字节为判断条件 |
| | | 0x01 | 2 | (Xs,Ys) | |
| | | 0x03 | 2 | (Xe,Ye) | |
| 0x000B | 圆弧 | 0x00 | 1 | Color0 | 圆弧显示 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 显示 | 0x01 | 2 | (X,Y) 0 | 圆心（X，Y）坐标，X坐标高字节为判断条件 | |
| | | 0x03 | 1 | RAD0 | 半径 | |
| | | 0x04 | 1 | DEG_S0 | 起始角度，单位 0.5°，0-720 | |
| | | 0x05 | 1 | DEG_E0 | 终止角度，单位 0.5°，0-720 | |
| 0x000C | 字符显示 | 0x00 | 1 | Color0 | 字符显示颜色 | |
| | | 0x01 | 2 | (X,Y) 0 | 字符显示位置，字符左上角坐标，X 坐标高字节为判断条件 | |
| | | 0x03H | 0.5 | Lib_ID | 字库位置 | |
| | | 0x03L | 0.5 | En_Mode | 字符编码模式：0=8bit 1=GB2312 2=GBK 3=BIG5 4=SJIS 5=UNICODE | |
| | | 0x04H | 0.5 | X_Dots | 字符 X 方向点阵数 | |
| | | 0x04L | 0.5 | Y_Dots | 字符 Y 方向点阵数 | |
| | | 0x05 | 1 | Text0 | 字符数据， 8bit 编码，对仅高字节有效<br>当编码方式为 01-04 时，如果字符数据为 ASCII 字符，将自动使用 0#预装字库显示 | |
| 0x000D | 矩形区域XOR | 0x00 | 2 | (x，y) s | 矩形区域左上角坐标，x坐标高字节为判断条件 | |
| | | 0x02 | 2 | (x,y)e | 矩形区域左下角坐标 | |
| | | 0x04 | 1 | Color | 矩形区域XOR的颜色，0xFFFF将进行反色操作 | |
| 0x000E | 双色位图显示 | 0x00 | 2 | (x,y)s | 位图显示矩形区域左上角坐标，x坐标高字节为判断条件 | |
| | | 0x02 | 1 | X_Dots | 位图X方向点阵数 | |
| | | 0x03 | 1 | Y_Dots | 位图Y方向点阵数 | |
| | | 0x04 | 1 | Color1 | “1”bit对应的显示颜色 | |
| | | 0x05 | 1 | Color0 | “0”bit对应的显示颜色：如果设置color0和color1相同，表示“0”bit不需要显示，直接跳过 | |
| | | 0x06 | N | Data_Pack | 显示数据，MSB方式；为方便用户读写数据，每行数据必须对齐到一个字，即下一行的数据总是从一个新数据字（word）开始 | |
| 0x000F | 位图显示 | 0x00 | 2 | (x,y)s | 位图显示矩形区域左上角坐标，x坐标高字节为判断条件 | |
| | | 0x02 | 1 | X_Dots | 位图 X 方向点阵数目 | 受变量空间大小限制，最大显示位图为 196*146（4:3）或 226*126（16:9） |
| | | 0x03 | 1 | Y_Dots | 位图 Y 方向点阵数目 | |
| | | 0x04 | N | Data_Pack | 显示数据，每个像素点一个字（MSB，565RGB 数据格式） | |
| 0x0010 | 区域放大一倍粘贴显示 | 0x00 | 2 | (x,y) | 放大一倍后图像粘贴在屏幕左上角坐标，X 高字节为判断条件 | 待放大区域位于放大后图像区域内时，必须右下角对齐。嵌套放大可以得到更大的放大倍数 |
| | | 0x02 | 2 | (x,y)s | 待放大矩形域左上角坐标 | |
| | | 0x04 | 2 | (x,y)e | 待放大矩形域右上角坐标 | |

判断条件：0xFF 绘图操作结束，0xFE 本次操作跳过（忽略）。

**基本图形显示举例（以 0x0006 图片区域剪切、粘贴指令为例）**

Step1 在界面上定义一个基本图形显示控件，变量存储地址设置为 0x1000；

| 区域范围设置 | |
|---|---|
| X坐标 | 0 |
| Y坐标 | 0 |
| 宽度 | 800 |
| 长度 | 480 |
| **变量属性** | |
| 名称定义 | 基本图形 |
| 描述指针(0x) | FFFF |
| 变量存储地址(0x) | 1000 |
| 虚线/点划线格式 | ☐ |

图 4-3-4 基本图形显示举例

Step2 串口向 0x1000 地址单元写入 0x0006 指令相关内容（把第三幅页面的(100,100)、(512,256)区域剪切粘贴到当前界面的(0,0)位置）。

图 4-3-5 基本图形指令举例

当显示切换到基本图形显示控件所在的页面时，VGUS 屏将执行剪切、粘贴。

坐标（340，237）与坐标（585，237）端点连线：

A5 5A 11 82 10 00 00 02 00 01 00 00 01 54 00 ED 02 49 00 ED

坐标（340，237）置点：

A5 5A 11 82 10 00 00 01 00 01 01 54 00 ED 00 00

左上角坐标（340，237）右下角坐标（585，340）矩形：

A5 5A 11 82 10 00 00 03 00 01 01 54 00 ED 02 49 01 54 00 00

圆心坐标（470，325），半径为 50 整圆：

A5 5A 0F 82 10 00 00 05 00 01 01 D6 01 45 00 32 00 00

1 号界面左上角坐标（265，64）右下角坐标（785，256）剪切粘贴到当前界面坐标（265，64）：

A5 5A 15 82 10 00 00 06 00 01 00 01 01 09 00 40 03 11 01 00 01 09 00 40

端点坐标（80，80）（128，128）线段显示：

A5 5A 11 82 10 00 00 0A 00 01 00 00 00 50 00 50 00 80 00 80

说明：上面所有指令中 00 01 指最大数据包数目，即执行画图指令的次数，如画一条线段、一个矩形框、一个圆设置为 00 01，画两条线段、两个矩形框、两个圆设置为 00 02。

### 4.3.3 列表显示（表格）

把按照二维数组定义的数据用**表格**分栏显示出来。

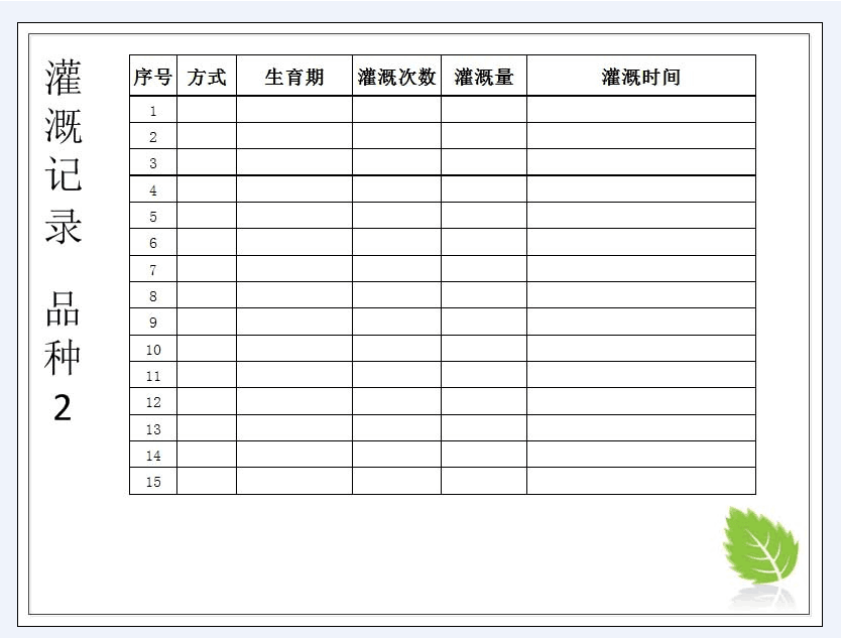



图 4-3-6 列表显示举例

| 变量属性 | |
|---|---|
| 名称定义 | 列表显示 |
| 描述指针(0x) | FFFF |
| 变量存储地址(0x) | 0500 |
| 列数 | 6 |
| 行数 | 4 |
| 起始显示列 | 1 |
| 首列不显示 | ☐ |
| 起始显示行 | 1 |
| 首行不显示 | ☐ |
| Unit_Data_Num | 0 |
| 编码方式 | 0x02=GBK |
| 字符间距自动调整 | ☐ |
| 直接变量显示 | ☐ |
| 显示边框 | ☐ |
| 边框颜色 | 0; 128; 0 |
| 文本颜色 | 192; 0; 192 |
| FONT0 ID | 0 |
| FONT1 ID | 34 |
| X方向点阵数 | 24 |
| Y方向点阵数 | 24 |

定义表格的列数、行数。

表格的内容可以显示文本，也可以显示变量。显示变量时参照备注【1】来确定要显示的变量的格式。

单元格的数据长度0x01-0xFF（单位：字），所有单元格的数据长度一样。

当设置为0时，表示可以按列设置单元格的数据长度，表格数据存储位置相应后延（列数/2）向上取整个字地址。
例如，变量地址为0x1000，列数为7，那么：0x1000-0x1003 依次存储了第 0-6 列的表格数据长度，其中 1003的低字节未使用。0x1004 地址开始存储表格内容。

图 4-3-7 列表显示属性设置

备注【1】

当不勾选“直接变量显示”，每个单元格数据内容的前两个字定义了表格数据格式，说明如下：

第一个字高字节选择数据类型Mode

0x00=整数(2 字节)，-32768 到 32767

0x01=长整数(4 字节)-2147483648 到 2147483647

0x02=*VP 高字节,无符号数0 到 255
0x03=*VP 低字节，无符号数0 到 255
0x04=超长整数(8 字节) -9223372036854775808 到 9223372036854775807
0x05=无符号整数(2 字节)0 到 65535
0x06=无符号长整数(4 字节)0 到 4294967295
0x10=时间格式 1，12:34:56BCD 码串
0x11=时间格式 2，12-34-56BCD 码串
0x12=时间格式 3，YYYY-MM-DD HH:MM:SS BCD 码串
0xFF=文本格式

第一个字低字节：

Mode=0x00-0x06定义了变量数据的定点显示格式，高 4bit 表示整数位数，低 4bit 表示小数位数。
Mode=0x10-0X11时间 BCD 码串的字节长度
Mode=其它无定义

第二个字定义单元格文本颜色。

如果表格实际内容短于 Unit_Data_Num 规定的长度时，使用 0xFFFF 做为单元格文本结束符。

对于特别大的表格，通过描述指针修改起始显示列、起始显示行可以很方便的实现表格的定位和拖动。

### 4.3.4 二维码

该控件实现将一段文字转换为二维码显示在串口屏上。把需要显示的二维码对应的文字、通过 0x82 指令写入变量存储地址单元中后，串口屏会根据写入的文字内容生成二维码、并按照指定位置显示出来。

当文字内容小于等于 154 字节时，二维码大小为 45x45 单元；当文字内容大于 154 字节时，二维码大小为 73x73 单元。每个单元所占实际物理像素点数量由图 4-3-8 中的 Unit_Pixels 定义。

首先在显示页面上添加一个二维码控件，设置属性中的变量存储地址，如图 4-3-8。然后把需要显示的二维码对应的文字、通过 0x82 指令写入变量存储地址单元中就可以了。VGUS 屏会根据写入的文字内容生成二维码、并按照指定位置显示出来。

需要注意的是：使用 0x82 指令发送文本时，需要在文本结束位置加上 0xFFFF 作为结束符。

字地址，范围 0000~FFFF，用于存储变量值，长度由文本结束标志 0xff 或者 0x00 指定（字地址一个单元包含 2 个字节），允许文本最长为 458 字节。使用 0x82 指令刷新显示内容时需要使用该地址。

| 变量属性 | |
|---|---|
| 名称定义 | 二维码0 |
| 变量存储地址(0x) | 0000 |
| 字长度 | 229 |
| **显示格式** | |
| Unit_Pixels | 4 |
| 描述指针(0x) | FFFF |

图 4-3-8 二维码属性设置

1 个汉字占用长度为 1，1 个 ASCII 码字符占用长度为 0.5，范围为 1-229 字。该长度仅用于 VGUS 开发工具自动分配变量存储地址和检查单元地址冲突。实际长度以文本结束标志为准，遇到文本结束标志 0xFFFF 后不再显示。

每个二维码单元像素所占的物理像素点阵大小，0x01-0x07。 设置 Unit_Pixels=4，那么每个单元像素将显示为 4*4 点阵大小。

例如：显示网站 www.viewtech.cn 生成的二维码，串口发送指令：

A5 5A 14 82 00 00 77 77 77 2E 76 69 65 77 74 65 63 68 2E 63 6E FF FF

A5 5A：帧头

14：指令字节长度，共 20 字节（不含帧头）

82：写变量存储器指令

00 00：变量地址，在控件属性中设置

77 77 77 2E 76 69 65 77 74 65 63 68 2E 63 6E：数据内容“www.viewtech.cn”的 ASCII 码。

FF FF：文本结束符

## 4.4 实时时钟 RTC 的显示

图 4-4-1 时钟显示举例

### 4.4.1 文本时钟

该控件用于实现图 4-4-1 中的文本时钟显示。时钟源可以选择串口屏自带 RTC 电路，也可以选择用户单片机提供的时间信息。对于没有 RTC 时钟功能的 VGUS 串口屏，必须把时钟源选择“用户单片机”模式（2018 年 11 月及以后固件版本支持）。

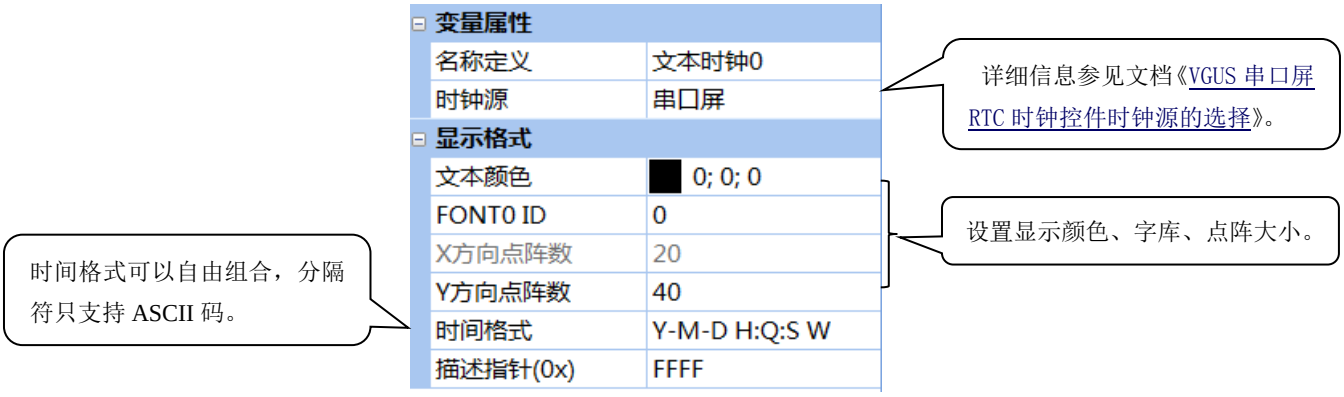



图 4-4-2 文本时钟属性设置

时间显示格式支持“年、月、日、时、分、秒、星期”七个时间单元，分别用表 5-4-1 的编码代替。可以在时间格式中任意组合这七个时间单元顺序、以及时间单元间的分隔符。

表 5-4-1 时间格式编码定义

| 说明 | 编码 | 显示格式 |
|---|---|---|
| 公历_年 | Y | 2000-2099 |
| 公历_月 | M | 01-12 |
| 公历_日 | D | 01-31 |
| 公历_小时 | H | 00-23 |
| 公历_分钟 | Q | 00-59 |
| 公历_秒 | S | 00-59 |
| 公历_星期 | W | SUN MON TUE WED THU FRI SAT |

如果需要使用中文分隔符，例如显示“2015 年 06 月 01 日”，需要设计三个文本时钟控件，分别用于显示年、月、日，并且把中文的年、月、日分隔符设计到背景图片上。

### 4.4.2 表盘时钟

该控件用于实现图 4-4-1 中的表盘时钟显示。时钟源可以选择串口屏自带 RTC 电路，也可以选择用户单片机提供的时间信息。对于没有 RTC 时钟功能的 VGUS 串口屏，必须把时钟源选择“用户单片机”模式（2018 年 11 月及以后固件版本支持）。

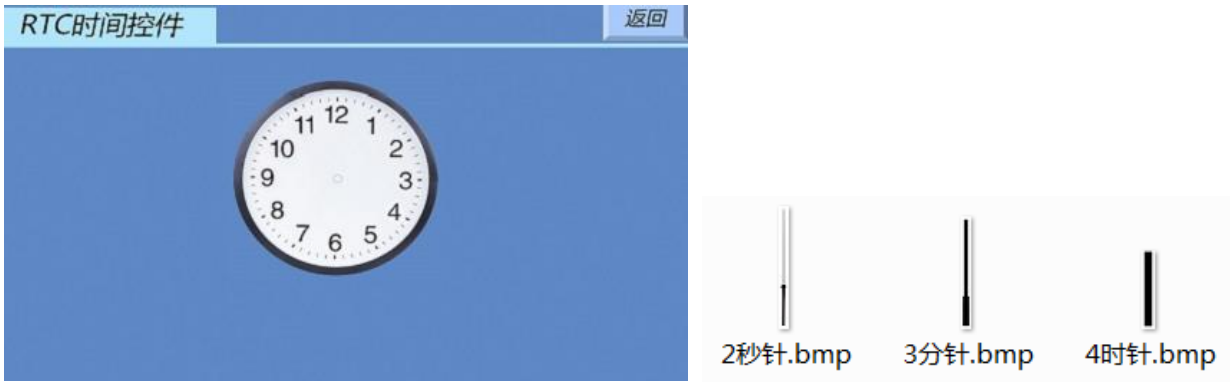



图 4-4-3 表盘时钟素材

在使用表盘时钟控件前，需要先准备一张表盘背景图片和三张图标（时针、分针、秒针），如上图所示。

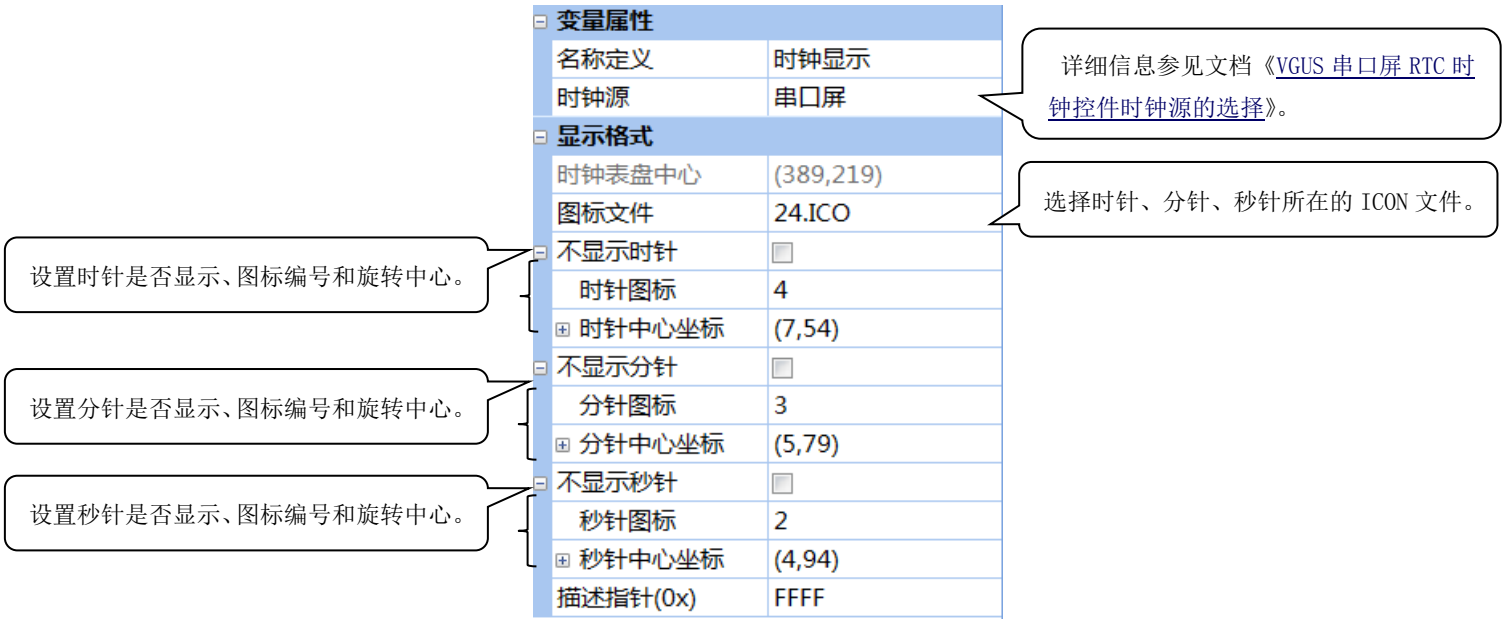



图 4-4-4 表盘时钟属性设置

### 4.4.3 变量时钟

变量时钟只能使用用户主板提供的时钟信息，通过串口把时间信息实时发送给串口屏，然后使用变量时钟控件来显示。

变量时钟只能显示文本形式的时钟。变量时钟分配了一段变量存储单元，单元长度通过 BCD 码字节数设定，单元功能用于存储单片机发送过来的“年、月、日、时、分、秒、星期”等时间信息，格式是 BCD 码格式。用户单片机将“年、月、日、时、分、秒、星期”等时间信息以 BCD 码格式、通过 0x82

指令写入变量存储单元。串口屏依次取出变量单元里面的 BCD 时间并按照设置好的显示位置、颜色、字库、点阵大小显示出来，字节与字节之间的分隔符客户可以设置（例如习惯年月日用中划线、时分秒用冒号）。显示一个字节的 BCD 码时间、取一个分隔符显示。

字地址，范围 0000~FFFF，用于存储 BCD 码格式的时钟，长度由 BCD 码字节数设定（字地址一个单元包含 2 个字节）。使用 0x82 指令刷新显示内容时需要使用该地址。用户单片机将“年、月、日、时、分、秒、星期”等时间信息以 BCD 码格式、通过 0x82 指令写入变量存储单元。

| 变量属性 | |
|---|---|
| 名称定义 | 时钟变量0 |
| 变量存储地址(0x) | 0000 |
| BCD码字节数 | 1 |
| **显示格式** | |
| 分隔符 | |
| 16进制分隔符 | |
| 文本颜色 | 0; 0; 0 |
| 字库位置 | 0 |
| X方向点阵数 | 16 |
| Y方向点阵数 | 32 |
| 描述指针(0x) | FFFF |

设置分隔符，每显示一个字节的 BCD 码数据，依次取一个分隔符。

以十六进制形式录入分隔符。

图 4-4-5 变量时钟属性设置

图 4-4-5 中的分隔符和 16 进制分隔符都是用来录入时间单元间的分隔符号，时间显示中的常用分隔符号只有中划线、冒号和空格，中划线用于分隔年月日、冒号用来分隔时分秒、空格用于分隔年月日和时分秒。ASCII 码中十六进制数 2D 是中划线、20 是空格、3A 是冒号。图 4-4-5 中时钟变量设置有 6 字节的 BCD 码时间数据、5 个分隔符号（2 个中划线+1 个空格+2 个冒号），假设变量存储单元里的 6 字节 BCD 数据依次是 15、09、01、09、20、25，那么显示出来的时间样式就是“15-09-01 09:20:25”。

分隔符还支持 2 个特殊编码 00 和 0D，只能通过 16 进制分隔符位置录入。00 不插入分隔符，实现 2 个 BCD 连到一起显示，常用与 4 位数的年份显示。0D 为换行显示，实现纵向排列时间显示，即 X=Xs，Y=Y+Font_X*2。例如设置变量时钟属性如图 4-4-6 所示，BCD 字节长度为 7 字节、6 个分隔符（00+2 个中划线+1 个空格+2 个冒号），假设变量存储单元里的 7 字节 BCD 是 20、15、09、01、09、20、25，那么显示出来的时间样式就是“2015-09-01 09:20:25”。

| | |
|---|---|
| Byte_Num | 7 |
| 编码字符串 | -- :: |
| 16进制录入 | 002D2D203A3A |

图 4-4-6 四位数的年份显示设置

例如：显示时间变量“08：30”，串口发送指令：

A5 5A 05 82 6F 00 08 30

说明：A5 5A：帧头

05：指令字节长度，82 6F 00 08 30 共 5 字节（不含帧头）

82：写变量存储器指令

6F 00：变量存储地址，在控件属性中设置

08 30：数据内容， 08 30 的 BCD 码

## 4.5 视频控件

该控件是在显示屏某一位置播放某一指定 AVI 视频。变量只有 2 个取值：一个为开始值，一个为停止值。当通过 0x82 指令向变量单元写入开始值时，视频开始播放；当写入停止值时，视频停止播放。

字地址，范围 0000~FFFF，用于存储变量，长度为一个单元（字地址一个单元包含 2 个字节）。变量只取 2 个值：一个为开始值，一个停止值。当通过 0x82 指令向该单元写入开始值时，视频开始播放；当写入停止值时，视频停止播放。

| 变量属性 | |
|---|---|
| 名称定义 | 视频播放0 |
| 变量存储地址(0x) | 4000 |
| 初始值 | 1 |
| **显示格式** | |
| 停止值 | 0 |
| 开始值 | 1 |
| 视频文件 | 0 |
| 播放结束时 | 0 |
| 重复次数 | 1 |
| 播放音量 | 63 |
| 描述指针(0x) | FFFF |

选择视频文件。

播放完毕后画面跳转页面。

图 4-5-1 视频控件属性设置

按照图 4-5-1 设置：

当向 4000 单元写入 1 后，将播放“0.AVI”文件，播放完后切换到 0 号页面。

发送指令：A5 5A 05 82 40 00 00 01

A5 5A：帧头

05：指令字节长度，82 40 00 00 01 共 5 字节

82：写变量存储器指令

40 00：设置的变量地址，如上图变量属性所示

00 01：设置的开始值。

当向 4000 单元写入 0 后，视频停止播放。

当向 4000 单元写入其它值时无效，不执行任何操作。

小技巧：使用视频控件实现开机自动播放视频“0.AVI”

如果把图 4-5-1 中的变量初始值设置为 1，就可以实现开机自动播放视频“0.AVI”。

关于视频播放的更详细信息请参考文档《VGUS 串口屏视频播放功能详解》。

# 5 触摸控件

## 5.0 触摸控件介绍

### 5.0.1 触摸控件汇总

触摸控件需要通过“VGUS开发工具-》触控配置工具栏”添加，触控配置工具栏各图标定义如图5-0-1所示。本章将逐一介绍各触控/键控按钮的使用方法、属性设置等。

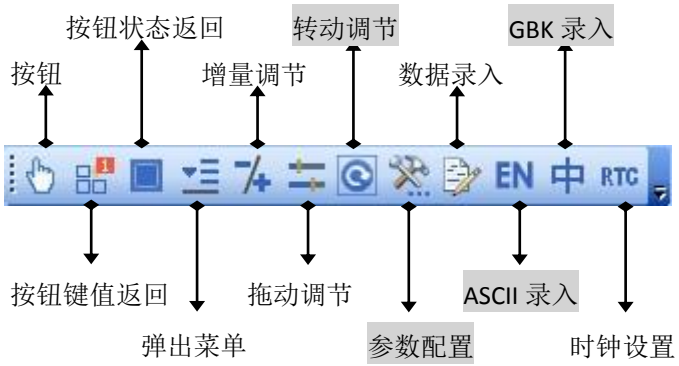



图 5-0-1 触控配置工具栏

表 5-0-1 触控/键控汇总表（***灰色背景部分为 SDWn 系列不支持的***）

| 序号 | 功能 | 说明 |
|---|---|---|
| 1 | 按钮 | “按钮”与“按钮键值返回”是两种基本按钮控件，都可以实现切换页面。 |
| 2 | 按钮键值返回 | “按钮”不能返回键值，常用于下文提到的“弹出菜单”、“数据、ASCII 与 GBK 录入”。 |
| 3 | 按钮状态返回 | “按钮键值返回”可以返回键值，按键按下后可以通过串口以 0x83 指令形式把键值发送给用户单片机。<br>“按钮状态返回”可以返回按钮的三种按压状态“按下、长按、抬起” 。 |
| 4 | 弹出菜单 | 利用“按钮”控件，实现典型的弹出菜单、下拉菜单功能。 |
| 5 | 数据录入 | 先要利用“按钮”控件制作录入键盘。<br>数据录入支持录入整数、定点小数等各种数据到指定变量存储空间。 |
| 6 | ASCII 录入、GBK 录入、RTC 设置 | 文本方式录入各种字符，录入过程支持光标移动、编辑；直接支持 ASCII 字符、GBK 中文、繁体注音输入法录入；修改字库和 0 号字库可以支持所有类似 ASCII 字符的 8bit 编码文本录入。RTC 设置需要完整录入公历年月日时分秒。 |
| 7 | 增量调节 | 点击按钮，对指定变量进行+/-操作，可设置步长和上下限。<br>设置 0-1 范围循环调节可以实现栏目复选框功能 |
| 8 | 拖动调节 | 拖拉滑块实现变量数据录入，可设置刻度范围。 |
| 9 | 转动调节 | 转动旋钮实现变量数据录入，可设置刻度范围。 |
| 10 | 硬件参数配置 | 提供了触摸屏改写寄存器空间的方法，来间接控制硬件。<br>比如把背光寄存器内容读取到变量，调节变量后再回写来调节背光亮度 |
| | | 点击触摸屏，把指定 VP 区域的数据发送到用户串口（COM1）。 |

### 5.0.2 触摸控件的典型属性

触摸控件的典型属性如图5-0-2所示，控件属性分为区域范围设置、按钮属性、变量属性和控件属性四大类。区域范围设置、按钮属性和变量属性对所有控件都是类似，而控件属性对不同触摸控件可能完全不同。

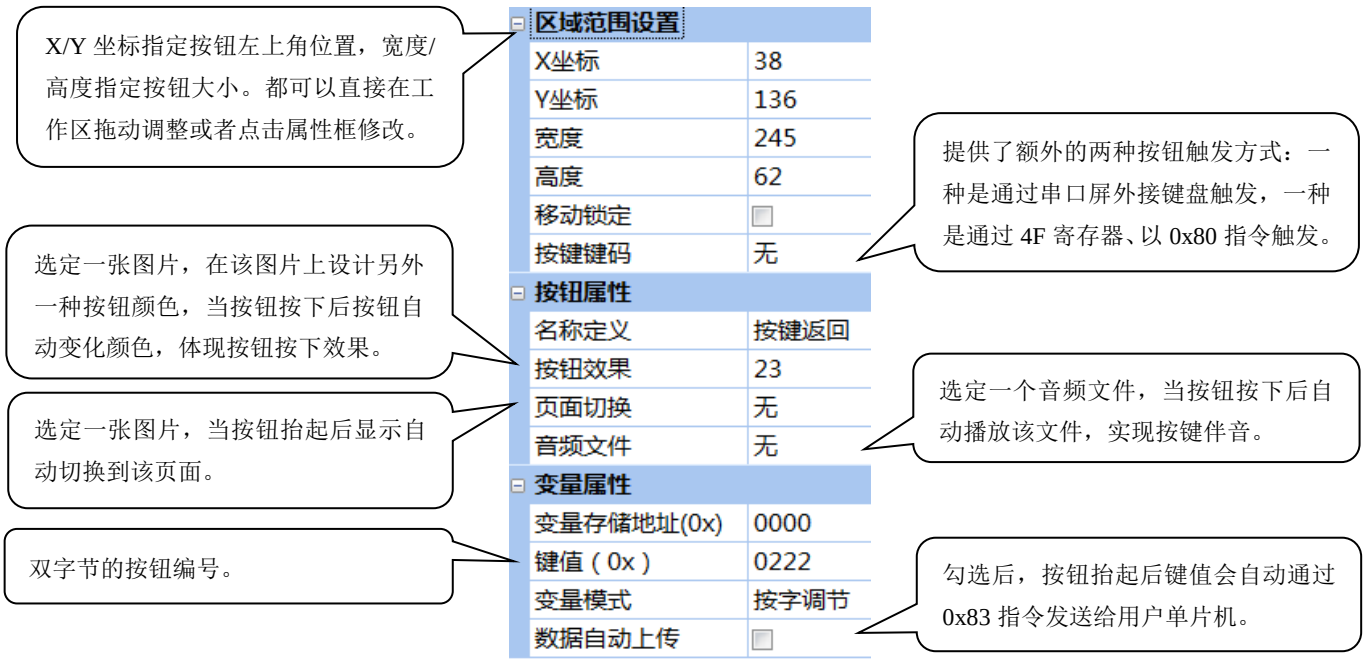



图 5-0-2 按钮区域范围设置属性

区域范围设置用于设置按钮的大小和位置。

按钮属性用于设置按钮按下反色显示效果、按钮按下后页面切换到那张图片显示、以及按钮按下播放哪个音频文件。

变量属性用于设置变量存储地址、键值、以及按钮按下后是否发送键值等。学习触摸控件的用法，务必了解清楚变量存储地址的作用和用法、以及控件属性的功能。变量存储单元用于存储键值，键值是双字节的按钮编号。在把键值保存到变量存储单元之前，先要把键值按照变量模式调整，0x83指令发送的键值也是调整之后的。勾选数据自动上传，按钮抬起后键值会自动通过0x83指令发送给用户单片机。

触摸控件的区域范围设置、按钮属性、变量属性用法基本一样，后续不再赘述。鼠标点击指定属性，在属性设置窗口下方会显示关于该属性的详细介绍。

### 5.0.3 按钮的基本触发方式

按钮的基本触发方式是通过点击触摸屏按钮的有效区域（由 X/Y 坐标和宽度/高度设置）。此外通过按键键码可以提供额外的两种按钮触发方式：一种是通过串口屏外接键盘触发（详细信息参考文档：《电阻触摸屏失效分析与 VGUS 串口屏外接键盘使用方法》），一种是通过 4F 寄存器、以 0x80 指令方式触发（通常以弹出菜单方式用于设备状态报警显示、或者通过用户单片机主板键盘触发）。

注意区分按钮键码与键值。键码是用来触发按钮的；键值是指按钮按下后，通过串口发送给用户单片机的按钮编号。

## 5.1 基本按钮

VGUS 屏设置有三种基本按钮控件：按钮、按钮键值返回、按钮状态返回，三者之间各有异同。三种按钮都可以实现页面切换功能，但是“按钮”不能返回键值，“按钮键值返回”可以返回键值，“按钮状态返回”可以返回按钮的三种按压状态：按下、持续按下和抬起。

### 5.1.1 按钮

按钮可以实现基本的页面切换，但不能通过串口返回键值，因为按钮没有分配变量存储地址。按钮一般用于弹出菜单和键盘录入、以及不需要返回按钮键值的场合。

第一步：准备 2 张按钮的图片素材，坐标位置重叠。一张是正常显示效果，一张是按下显示效果。

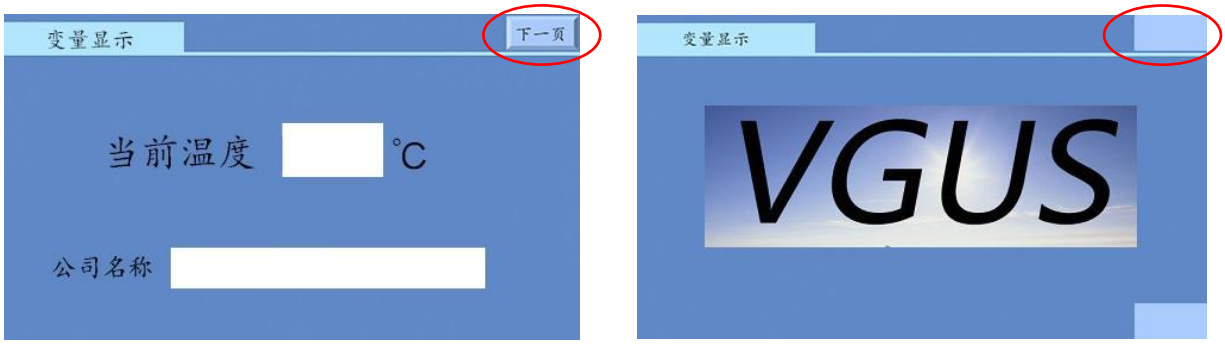



图 5-1-1 按钮效果图片素材

第二步：添加按钮并设置属性

| 区域范围设置 | |
|---|---|
| X坐标 | 21 |
| Y坐标 | 87 |
| 宽度 | 244 |
| 高度 | 102 |
| 移动锁定 | ☐ |
| 按键键码 | 无 |
| **按钮属性** | |
| 名称定义 | 基本触控 |
| 按钮效果 | 1 |
| 页面切换 | 2 |
| 音频文件 | 无 |
| **变量属性** | |
| 键值（0x） | 0000 |

双字节的按钮编号，用于弹出菜单和键盘录入。用于弹出菜单中的按钮时高字节必须为 0，范围 0000~00FF，其中 00FF 表示不返回键值；用于键盘录入时，点击"..."正确设置键值。

图 5-1-2 按钮属性设置

区域范围设置、按钮属性的用法参见图 5-0-2。

变量属性只有一栏键值，没有分配变量存储地址，因此不能通过 0x83 指令返回键值；在“弹出菜单”控件中，弹出窗口中设计的“按钮”控件，会利用弹出菜单控件的变量存储地址来存储“按钮”控件键值，同时也能实现通过 0x83 指令返回键值。

### 5.1.2 按钮键值返回

相对“按钮”控件而言，“按钮键值返回”控件多出了键值自动上传功能。当按键抬起后，可以通过串口以 0x83 指令把键值发送给用户单片机。 按钮键值返回属性设置如图 5-1-3 所示。

字地址，范围 0000~FFFF，用于存储键值，长度为一个单元（字地址一个单元包含 2 个字节）。按钮抬起后相应键值会保存到该单元中。通常把所有按钮都分配相同的变量存储地址，通过键值来区分按钮。使用 0x83 指令发送键值时会使用该地址。

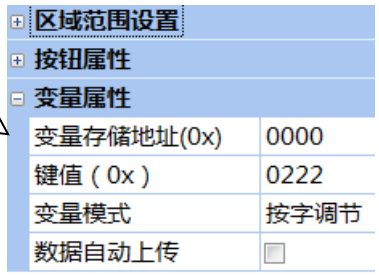



图 5-1-3 按钮键值返回属性设置

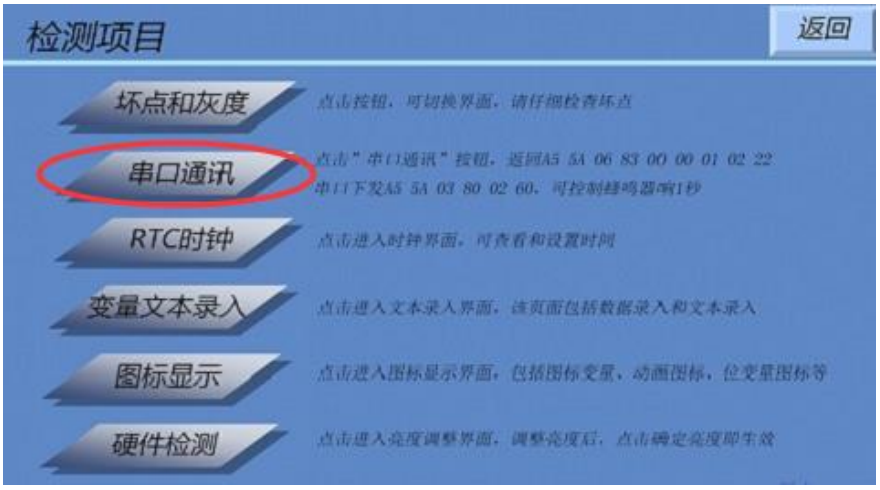



图 5-1-4 按钮键值返回举例

例如，在图 5-1-4“串口通讯”位置添加有按钮键值返回按钮，属性设置如图 5-1-3，当点击“串口通讯”按钮后，串口返回指令：A5 5A 06 83 00 00 01 02 22

说明：A5 5A：帧头

06：指令字节长度，83 00 00 01 02 22 共 6 字节（不含帧头）

83：读变量存储器指令

00 00：变量存储器地址

01：数据字长度，02 22 共 1 字长

02 22：数据内容，设置的键值

### 5.1.3 按钮状态返回

持续按下触摸按钮后，如图 5-1-5 所示，分为三个状态：第一次按压状态、持续按压状态和抬起状态。在有些应用场合用户单片机可能需要识别这些状态，可以通过按钮状态返回控件实现。

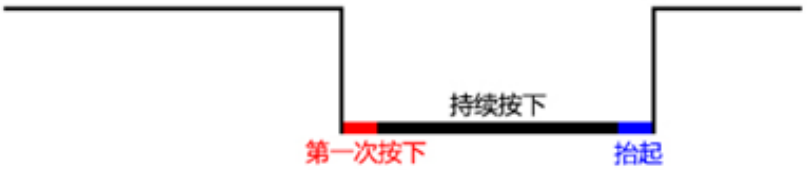



图 5-1-5 按钮持续按压的三种状态

相对“按钮键值返回”控件而言，“按钮状态返回”控件可以返回按钮按下的三种状态，并且返回值用户可以灵活自定义。按钮状态返回按钮属性如图 5-1-6 所示，分别定义了三种状态下的按压模式、S 变量存储地址、S 变量数据长度、S 变量初始值、T 变量存储地址、T 变量初始值。

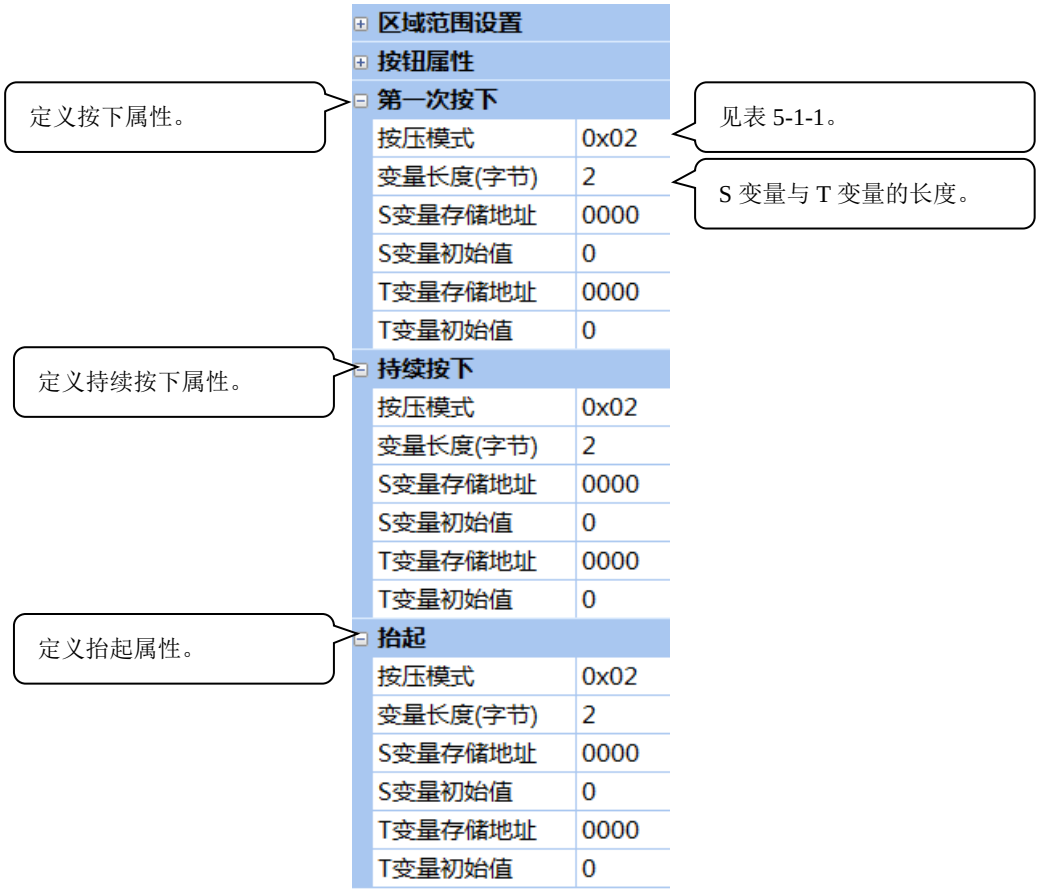



图 5-1-6 按钮状态返回属性设置

表 5-1-1 按压模式

| 按压模式 | 说明 |
|---|---|
| 0x00 | 不做任何处理 |
| 0x01 | 读取 S 地址处的指定长度数据写到 T 地址里（数据长度必须为偶数字节） |
| 0x02 | 把 S 变量数据通过串口发送给用户单片机 |
| 0x03 | 读取 S 地址处的指定长度数据写到 T 地址对应的寄存器里（T 地址高字节必须为 0x00） |

当把按压模式都设置为 0x02 时，三种状态下串口屏都会将各自的 S 变量数据通过串口发送出去；其中持续按压状态下，每一个 VGUS 刷新周期（约 100 毫秒）都会重复发送一次。向 S 变量中写入特定的初始值，用户单片机根据接收到的数据，就可以识别出持续按钮的各个状态。

需要注意的是，在把 S 变量数据通过串口发送的时候，是不加帧头的，为了统一指令格式，可以把帧头信息放到 S 变量里面。例如 S 变量地址是 0x0110、长度为 2 个字节、初始值是 0x6666，那可以把 S 变量长度设置为 9 个字节，初始值设置为“a5 5a 06 83 01 10 01 66 66”，这样用户单片机接收到的数据就符合统一指令格式。

0x01、0x03 两种按压模式使用较少。

## 5.2 弹出菜单

利用“按钮”控件，实现典型的弹出菜单、下拉菜单设计。点击“弹出菜单”按钮后会弹出一个新窗口，新窗口上可以设计“按钮”控件，按钮键值可以通过弹出菜单的变量存储地址返回。点击弹出菜单窗口上面任意一个按钮，弹出窗口会自动消失。下拉菜单也可以使用本控件设计。

**第一步：在弹出菜单窗口上添加按钮并设置键值**

首先要把弹出菜单窗口保存成一张图片。然后在该弹出菜单窗口有效区域里，添加需要的按钮（见 5.1.1 节内容）并设置好按钮键值（取值范围 0000 到 00fe，00ff 专用于取消按钮）。**弹出菜单上的按钮必须设置键值。**

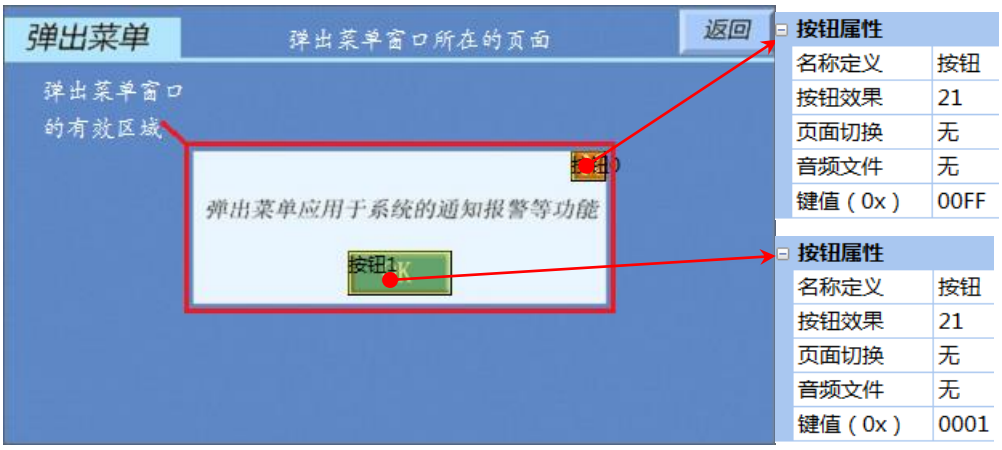



图 5-2-1 在弹出菜单窗口上添加按钮

弹出菜单窗口上设计了 2 个按钮，一个 OK 按钮和一个取消按钮。OK 按钮的键值设置为 0001，取消按钮的键值设置为 00ff。当弹出菜单窗口弹出后，点击上面任意一个按钮，弹出窗口会自动消失。

当点击 OK 按钮时，按钮键值 0001 会写入弹出菜单变量单元里，变量单元地址见 5.2.2 弹出菜单按钮属性设置，键值也可以通过串口指令 0x83 发送给用户单片机。取消按钮键值设置为 0x00ff，点击取消按钮后不会把键值写入变量单元里，也不会通过 0x83 指令发送键值给用户单片机。

**第二步：添加弹出菜单按钮并设置属性**

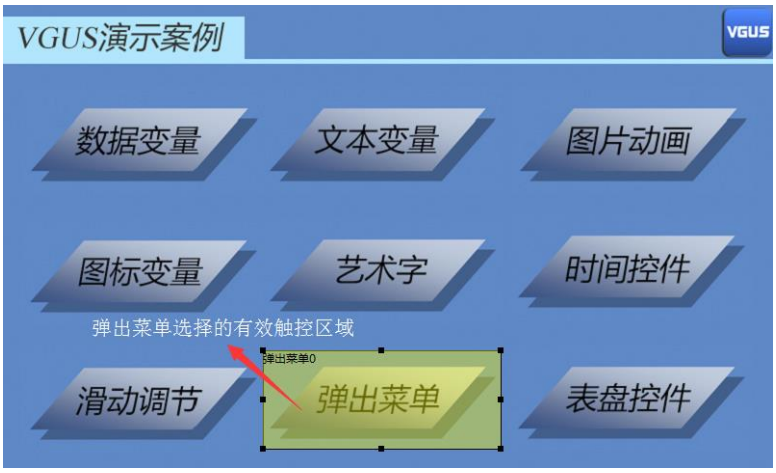



图 5-2-2 添加弹出菜单按钮

上图为添加一个弹出菜单按钮。一般来讲，弹出菜单是用于系统报警提示用，不会通过按钮来触发的。

所以实际使用中，通常会把该图中的弹出菜单按钮隐藏起来（触摸区域做的很小、放到角落位置、并且不要在背景图片上体现按钮），设置按键键码、通过 0x80 指令写 4F 寄存器来触发。

字地址，范围 0000~FFFF，用于存储键值，占用一个单元（字地址一个单元包含 2 个字节）。弹出菜单页面上的按钮抬起后，相应按钮键值会保存到该单元中。使用 0x83 指令上传键值时会使用该地址。

| 变量属性 | |
|---|---|
| 变量存储地址(0x) | 0200 |
| 变量模式 | 按字调节 |
| 数据自动上传 | ☑ |
| **控件属性** | |
| 弹出菜单设置 | Click to set |
| 所在页面 | 20 |
| ⊞ 菜单区域 | (196,157) (622,326) |
| ⊞ 显示位置 | 170,160 |

选择弹出菜单所在的页面，并拖动选择弹出菜单窗口区域。

选择弹出菜单在当前页面上的显示位置（窗口左上角坐标）。

图 5-2-3 弹出菜单按钮属性设置

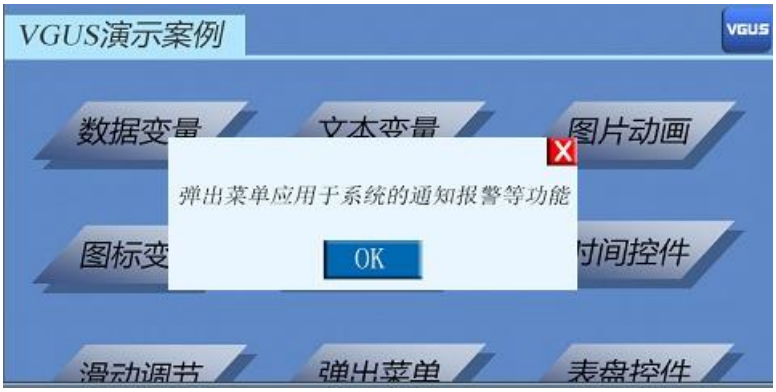



图 5-2-4 弹出菜单执行效果

例如：设置“弹出菜单”变量存储地址为 0200，在弹出菜单页面上添加“OK”按钮的键值为 0001。点击“弹出菜单”按钮会弹出菜单窗口，点击菜单上的“OK”按钮，会将按钮的键值 0001 返回到地址 0200 单元里，弹出菜单窗口消失，串口返回指令：

A5 5A 06 83 02 00 01 00 01

说明：A5 5A：帧头

06：指令字节长度，83 02 00 01 00 01 共 6 字节（不含帧头）

83：读变量存储器指令

02 00：变量地址，在弹出菜单控件属性中设置

01：数据字长度，00 01 共 1 字长

00 01：数据内容，设置的按钮键值 00 01

## 5.3 键盘录入

VGUS 屏支持数据录入、英文录入、中文全拼录入、时钟设置等。在使用录入控件之前，必须先制作满足功能需要的录入键盘。

### 5.3.1 制作键盘

第一步：准备素材。

键盘有效区域分为两个部分：录入过程中的文本显示窗口、和按钮区域。把所有需要用到的按钮做成图片，按照实际使用的效果排列好。一张正常显示的图片，一张按钮按下时显示的图片。

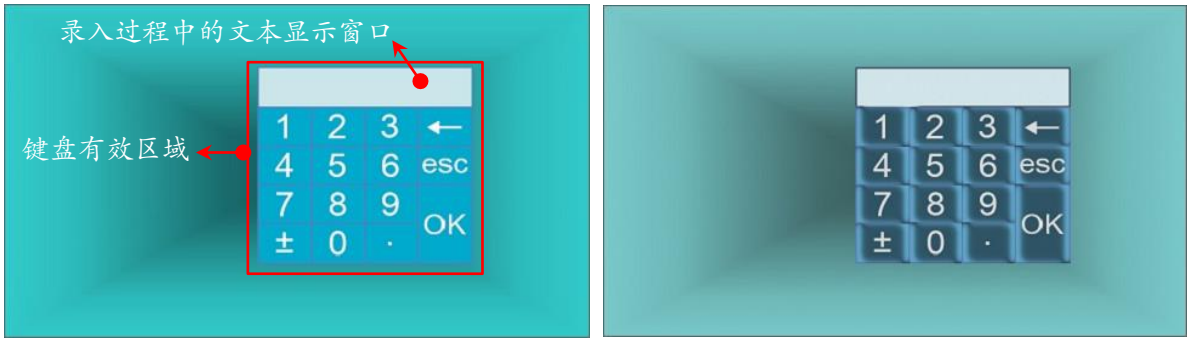



图 5-3-1 数字小键盘素材

第二步：添加按钮并设置键值。

参照 5.1.1 节内容，添加上图中的 15 个按钮并设置好属性。此处按钮键值不能任意写入数字，必须首先将鼠标指向键值属性并点击，键值属性右侧会出现图标“…”，点击该图标展开键值选择窗口，根据每个按键的实际功能做相应选择，选择后系统会自动正确提供该按钮的键值。

图 5-3-2 数字键盘按钮 1 的键值设置过程

### 5.3.2 数据录入

点击“数据录入”按钮后，会弹出录入键盘实现数据录入的功能。在学习数据录入之前，需要先掌握数据变量显示一节的内容(4.1.1 节)。

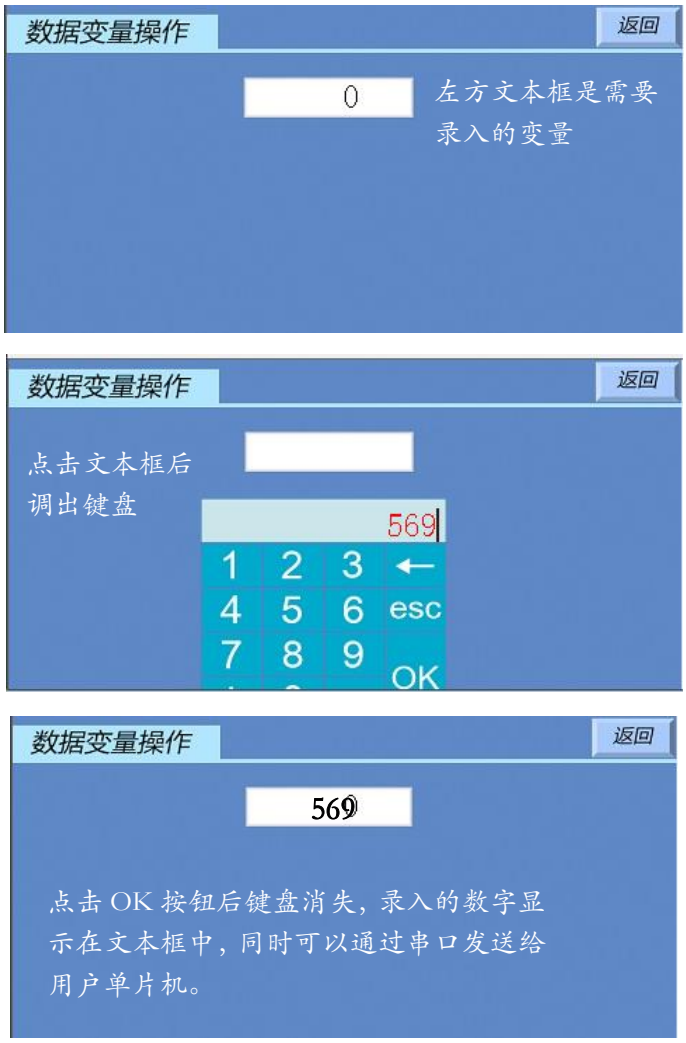



图 5-3-3 数据录入应用举例

上面三张图片介绍了我们将要实现的数据录入功能。为了实现上述功能，我们需要在文本框位置添加两个控件：一个是数据变量显示控件，一个是数据录入控件。

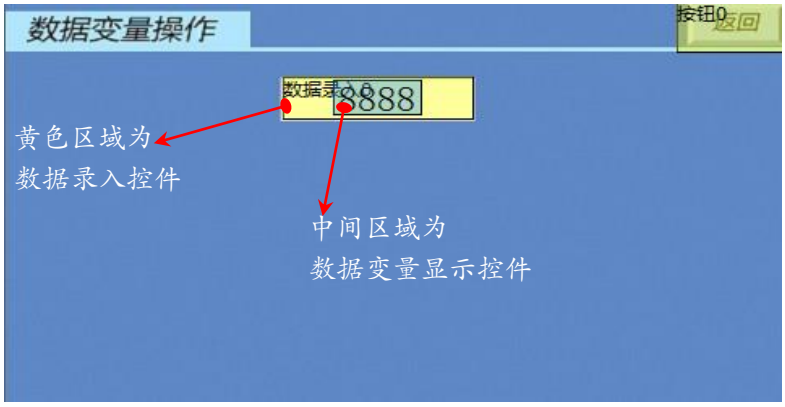



图 5-3-4 数据录入控件设计

数据变量显示控件属性设置参见 4.1.1 节，数据变量录入控件如图 5-3-5 所示。

字地址，范围 0000~FFFF，用于存储录入数据，长度由变量类型设定（字地址一个单元包含 2 个字节）。使用 0x83 指令上传录入数据时会使用该地址。

选择录入键盘所在的页面，并拖动选择键盘有效区域，以及键盘在当前页面上的显示位置（左上角坐标）。

设置录入过程中文本显示窗口位置和数字显示格式。

| | |
|---|---|
| **变量属性** | |
| 变量存储地址(0x) | 0020 |
| 变量类型 | 整数（字） |
| 整数位数 | 4 |
| 小数位数 | 0 |
| 数据自动上传 | ☑ |
| 变量反色显示 | ☐ |
| 启用范围限制 | ☐ |
| **键盘属性** | |
| 键盘在当前页面 | ☐ |
| 键盘设置 | Click to set |
| 所在页面 | 6 |
| 键盘区域 | (345,89) (638,370) |
| 显示位置 | 221,164 |
| **键盘文本窗口属性** | |
| 显示位置 | 497,172 |
| 文本颜色 | 0; 0; 0 |
| 字库位置 | 0 |
| X方向点阵数 | 16 |
| Y方向点阵数 | 32 |
| 光标颜色 | 黑色 |
| 输入显示方式 | 正常显示 |

定义数据格式，数据写入变量存储单元前，会按照数据格式化整。VGUS 屏中没有实际意义的小数，实际存储都是整数形式的，小数只是整数中按照指定位置加小数点。比如“10.1”，实际存储、串口通讯都处理为 0x0065(十进制为 101)，而显示的时候根据小数位数为 1、显示为“10.1”。

指录入过程中的数据显示窗口位置（如图 5-3-1）。一般会把录入过程显示框与键盘做在一起，**所以需要先设置键盘属性**，然后才能看到键盘显示框位置，点击拖动选取。

图 5-3-5 数据录入按钮属性设置

按钮属性栏里面有个变量“反色显示属性”，勾选后，将对相应的存储单元内变量反色显示。一般用于页面上有多个录入变量时，用此功能将当前录入的变量反色显示，以示区分。

如图 5-3-3 所示，如希望录入完成后，把录入的数据直接显示在文本框中，需要要图 5-4-4 中的数据变量控件和数据录入控件设置相同的变量存储地址。

例如，按照图 5-3-5 设置属性，设置录入数据整数位数为 4 位，点击“数据录入”按钮输入“1234”，点击“OK”，串口返回指令：

A5 5A 06 83 00 20 01 04 D2

说明：A5 5A：帧头

06：指令字节长度，83 00 20 01 04 D2 共 6 字节（不含帧头）

83：读变量存储器指令

00 20：变量地址，在控件属性中设置

01：数据字长度，04 D2 共 1 字长

04 D2：数据内容，1234 的十六进制数据

### 5.3.3 英文ASCII录入

点击“ASCII 录入”按钮后，会弹出录入键盘实现 ASCII 码字符录入。在学习 ASCII 录入之前，需要先掌握文本变量显示一节的内容(4.1.2 节)。

图 5-3-6 ASCII 录入举例

字地址，范围 0000~FFFF，用于存储录入的 ASCII 码字符，长度由字节长度设定（字地址一个单元包含 2 个字节），实际占用地址单元个数=字长度+2（一个单元的输入状态和一个单元的结束符）。使用 0x83 指令上传录入文本会使用该地址。

勾选表示要返回状态，状态信息参见备注1，不勾选表示不返回。

指录入过程中的文本显示窗口位置。一般会把录入过程显示框与键盘做在一起，**所以需要先设置下面的键盘属性**，然后才能看到键盘显示框位置，点击拖动选取。

| | |
|---|---|
| **变量属性** | |
| 变量存储地址(0x) | 0100 |
| 字长度 | 40 |
| 数据自动上传 | ☑ |
| 输入状态返回 | ☑ |
| 录入模式 | 重新录入 |
| **键盘属性** | |
| 键盘在当前页面 | ☐ |
| 键盘设置 | Click to set |
| 所在页面 | 8 |
| 键盘区域 | (108,102) (723,429) |
| 显示位置 | 68,141 |
| **键盘文本窗口属性** | |
| 文本显示区域 | (78,153) (676,201) |
| 文本颜色 | 0; 0; 0 |
| 字库位置 | 0 |
| X方向点阵数 | 12 |
| Y方向点阵数 | 24 |
| 光标颜色 | 黑色 |
| 输入过程显示方式 | 正常显示 |

设定录入 ASCII 码字符串的长度，范围 1-123，2 个 ASCII 码字符占用一个字。字符保存到指定地址时，自动在字符串结束位置加上 0xFFFF 作为结束符，输入结束上传到串口的数据也会加上 0xFFFF。结束符会占用一个变量存储单元，分配变量存储地址时需要注意，其它变量不能占用该单元地址。

设置录入过程中的文本显示位置和显示格式。

图 5-3-7 ASCII 录入按钮属性设置

**备注1：**勾选输入状态返回后，串口屏将把当前录入状态（0x00为录入中、0x5a为录入结束）和录入的有效数据长度（单位为字节）写在变量存储地址的前一个单元里。用户可以用0x83指令去读取录入状态和有效数据长度。例如上例中，变量存储地址是0x0100，因此会占用0x00ff单元，其中高字节用于存储录入状态、低字节用于存储录入的有效数据长度。在该例中因为占用了0x00ff单元，所以其它变量不能再使用。

ASCII录入实际占用的变量存储单元=字长度+2。额外占用的2个单元一个在变量存储位置前（状态返回）、一个在变量存储位置后（结束符）。在分配变量单元地址时要注意，其它变量地址不能重叠。

### 5.3.4 中文GBK录入

点击“GBK 录入”按钮后，会弹出录入键盘实现中英文字符录入。在学习之前，需要先掌握文本变量显示 (4.1.2 节)。首先我们看一款经典中文全拼录入键盘的界面设计。拼音窗口上部显示录入的拼音、下部显示检索到的汉字，文本窗口显示已经录入的汉字。拼音窗口与文本窗口中的汉字都必须使用 GBK 字库，需要在属性窗口分别设置字库、点阵大小等显示格式。

在 GBK 录入过程中，可以用 enter 健直接把拼音转为字母录入，实现 ASCII 码录入。拼音窗口和文本窗口中用到的 ASCII 码字符一律默认使用 0 号字库，点阵大小与中文 GBK 字库一致。

图 5-3-8 GBK 录入键盘举例

字地址，范围 0000~FFFF，用于存储录入中英文字符，长度由字符串长度设定（字地址一个单元包含 2 个字节），实际占用地址单元个数=字长度+2（一个单元的输入状态和一个单元的结束符）。使用 0x83 指令上传录入文本时会使用该地址。

勾选表示要返回状态，状态信息参见备注【1】，不勾选这表示不返回。

设置文本窗口的显示位置、显示格式。必须先设置键盘属性。

设置拼音窗口显示位置、显示格式。必须先设置键盘属性。

设置拼音窗口中拼音字母与检索到的汉字排列关系。

| | |
|---|---|
| **变量属性** | |
| 变量存储地址(0x) | 0140 |
| 字长度 | 40 |
| 数据自动上传 | ☑ |
| 输入状态返回 | ☑ |
| 录入模式 | 重新录入 |
| **键盘属性** | |
| 键盘在当前页面 | ☐ |
| 键盘设置 | Click to set |
| 所在页面 | 8 |
| 键盘区域 | (109,103) (723,429) |
| 显示位置 | 63,148 |
| **键盘文本窗口属性** | |
| 文本显示区域 | (342,154) (669,212) |
| 文本字库位置 | 66 |
| 文本颜色 | 0; 0; 0 |
| Y方向点阵大小 | 24 |
| **键盘拼音窗口属性** | |
| 拼音显示位置 | 71,154 |
| 拼音字库位置 | 66 |
| 拼音文本颜色 | 255; 0; 0 |
| Y方向点阵大小 | 24 |
| 拼音显示方式 | 上边 |
| 光标颜色 | 黑色 |
| 显示间距 | 1 |

设定录入字符串的长度，范围 1-123，1 个汉字占用 1 个字、2 个 ASCII 码占用 1 个字。文本保存到指定地址时，自动在字符串结束位置加上 0xFFFF 作为结束符，输入结束上传到串口的数据也会加上 0xFFFF。结束符会占用一个变量存储单元，分配变量存储地址时需要注意，其它变量不能占用该单元地址。

设置文本窗口汉字显示的字库、点阵大小和颜色，字库必须是GBK字库。ASCII字母默认用0号字库。

设置拼音窗口汉字显示的字库、点阵大小和颜色，字库必须是GBK字库。ASCII字母默认用0号字库。

图 5-3-9 GBK 录入按钮属性设置

**备注1：**与ASCII录入一样，勾选输入状态返回后，串口屏将把当前录入状态（0x00为录入中、0x5a为录入结束）和录入的有效数据长度（单位为字节）写在变量存储地址的前一个单元里。用户可以用0x83指令去读取录入状态和有效数据长度。例如上例中，变量存储地址是0x0180，因此会占用0x017f单元，其中高字节用于存储录入状态、低字节用于存储录入的有效数据长度。在该例中因为占用了0x017f单元，所以其它变量不能再使用。

与ASCII录入一样，GBK录入实际占用的变量存储单元=字长度+2。额外占用的2个单元一个在变量存储位置前（状态返回）、一个在变量存储位置后（结束符）。在分配变量单元地址时要注意，其它变量不能再占用。

图 5-3-10 GBK 录入举例

例如：点击“GBK 录入”按钮输入 “武汉中显”，点击 “Enter” 按钮，串口返回指令：

A5 5A 0E 83 01 40 05 CE E4 BA BA D6 D0 CF D4 FF FF

说明：A5 5A：帧头

0E：指令字节长度，83 01 40 06 CE E4 BA BA D6 D0 CF D4 FF FF 共 14 字节（不含帧头）

83：读变量存储器指令

01 40：变量地址，在控件属性中设置

05：数据字长度，CE E4 BA BA D6 D0 CF D4 FF FF 共 5 字长

CE E4 BA BA D6 D0 CF D4 FF FF：数据内容，“武汉中显”的汉字内码 CE E4 BA BA D6 D0 CF D4，**最后 2 字节 FF FF 是录入结束的标志位，文本录入及 ASCII 录入时有结束标志位。**

注：中显预装的 0#字库包含 4*8-64*128 点阵的所有 ASCII 字符。

不使用触摸屏，使用键盘（0x4F 寄存器保存的键码）来做 GBK 录入时，必须用 0x01-0x08 键码来选择对应的汉字。

### 5.3.5 时钟设置

点击时钟设置按钮，会弹出录入键盘完成时间设置，设置的时间会自动保存到 0x20-0x26 寄存器中。时钟设置按钮仅用于有 RTC 功能的串口屏，也就是针对文本时钟和表盘时钟控件，用于校准年月日、时分秒。

由于“时钟设置”控件没有分配变量存储地址，因此不能通过 0x83 指令返回设置时间；设置时间是直接写入了 0x20-0x26 寄存器中，因此是通过 0x81 指令返回设置时间。

时钟设置控件属性与数据录入基本一样，如图 5-3-11。

| | |
|---|---|
| **变量属性** | |
| 数据自动上传 | ☑ |
| **键盘属性** | |
| 键盘在当前页面 | ☐ |
| 键盘设置 | Click to set |
| 所在页面 | 6 |
| 键盘区域 | (344,89) (637,370) |
| 显示位置 | 491,187 |
| **键盘文本窗口属性** | |
| 显示位置 | 776,193 |
| 文本颜色 | 0; 0; 0 |
| 字库位置 | 0 |
| X方向点阵数 | 14 |
| Y方向点阵数 | 28 |
| 光标颜色 | 黑色 |

图 5-3-11 时钟设置按钮属性

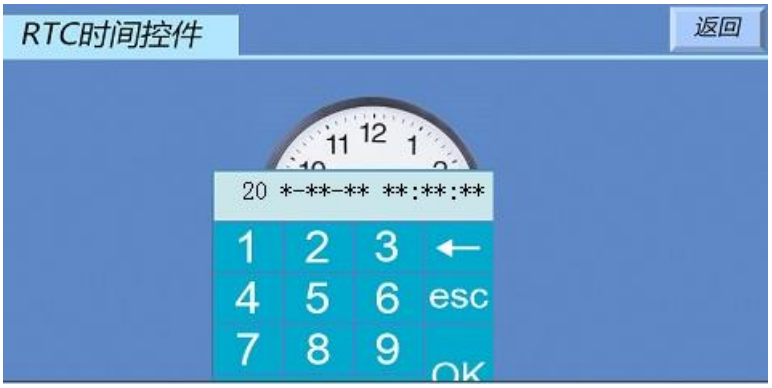



图 5-3-12 时钟设置举例

例如：启动时钟设置按钮后，输入时间 16-10-26 16：09：00，串口返回指令：

A5 5A 0A 81 20 07 16 10 26 03 16 09 00

说明：A5 5A：帧头

0A：指令字节长度，81 20 07 16 10 26 03 16 09 00 共 10 字节（不含帧头）

81：读寄存器指令

20：控制 RTC 的寄存器地址

07：数据字节长度，16 10 26 03 16 09 00 共 7 字节长

16 10 26 03 16 09 00：时间变量的 BCD 码

## 5.4 调节控件

### 5.4.1 增量调节

点击增量调节按钮，可以将指定变量、按照指定步长增加或者减小。

字地址，范围 0000~FFFF，用于存储变量，长度为一个单元（字地址一个单元包含 2 个字节）。增量调节按钮每按下一次，该变量会按照指定步长变化一次。使用 0x83 指令上传变量时会使用该地址。

变量值增加或减少超过设定的上下限的值时，继续按按钮将会停止调节，也可以选择循环调节。

| 变量属性 | |
|---|---|
| 变量存储地址(0x) | 6F01 |
| 变量模式 | 按字调节 |
| 数据自动上传 | ☑ |
| **控件属性** | |
| 调节方式 | -- |
| 调节步长 | 4 |
| 按键模式 | 按键连续调节 |
| 下限 | 0 |
| 上限 | 64 |
| 逾限处理方式 | 停止 |
| 长按快速调节 | ☑ |

按照指定调整方式，把变量值保存到变量存储单元。0x83 指令发送的变量值也是调整之后的。

按下按钮后变量增加（++）或减小（--）一个调节步长的值。

长按按钮下，可以选择连续调节（按钮按下 0.5S 后，每个刷新周期调节一次）或者单次调节。

图5-4-1 增量调节按钮属性设置

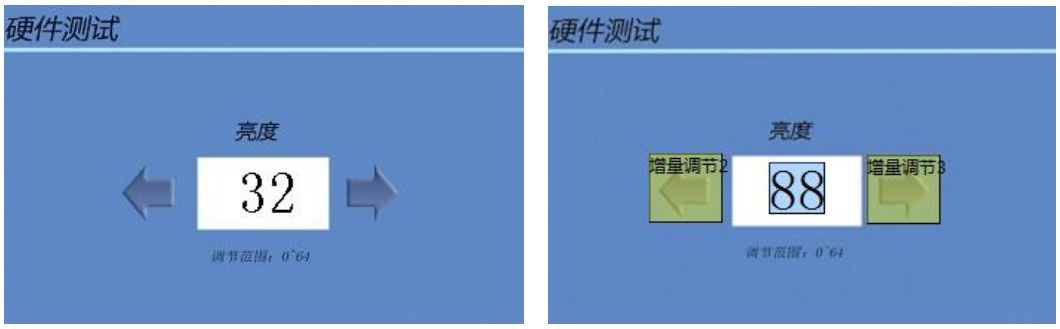



图5-4-2 增量调节举例

如图 5-4-2 左图，点击左箭头文本框数据减小，点击右箭头文本框数据增大。为实现上述增量调节功能，如上图右侧，需要设计两个增量调节按钮、一个数据变量显示。左箭头设置为－－、右箭头设置为＋＋，为了能让增量调节后的值能够显示在中间的数据变量位置，图中的 2 个调节按钮和数据变量必须定义相同的变量存储地址。

例如图 5-4-2，右箭头增量调节按钮属性设置如图 5-4-1 所示，连续点击右箭头按钮，串口返回指令：

A5 5A 06 83 6f 01 01 00 01　　A5 5A 06 83 6f 01 01 00 02　　……　　A5 5A 06 83 6f 01 01 00 09

说明：A5 5A：帧头

06：指令字节长度，83 6f 01 01 00 01 共 6 字节（不含帧头）

83：读变量存储器指令

6f 01：变量地址，在控件属性中设置

01：数据字长度，00 01 共 1 字长

00 01：数据内容，从设置的下限变化到上限，即从 00 01 连续变化到 00 09

当把属性中下限和上限分别设置为 0 和 1 时，遇限处理方式设置为循环调节，配合**变量图标**显示，可以方便的设计出栏目复选框功能（点击一下选中，再点击取消）。

当把属性设置中的下限和上限都设置为 0，遇限处理方式设置为停止，可以用于实现连续按键识别。按键按下 0.5 秒后，每个 VGUS 刷新周期（约 100 毫秒）都会发出串口数据。

### 5.4.2 拖动调节

按下并移动“拖动调节”按钮，可以将指定变量、按照设定的范围自动变化。

字地址，范围 0000~FFFF，用于存储变量，长度为一个单元（字地址一个单元包含 2 个字节）。按下拖动调节按钮后，系统会根据拖动调节按钮的当前位置自动计算变量值（根据起始值和终止值进行插值运算）。使用 0x83 指令上传变量时会使用该地址。

| 变量属性 | |
|---|---|
| 变量存储地址(0x) | 0180 |
| 变量模式 | 按字调节 |
| 数据自动上传 | ☑ |
| **控件属性** | |
| 拖动方向 | 横向拖动 |
| 起始值 | 0 |
| 终止值 | 1000 |

拖动范围的起始位置对应的变量值。

拖动范围的终止位置对应的变量值。

图5-4-3 拖动调节属性设置

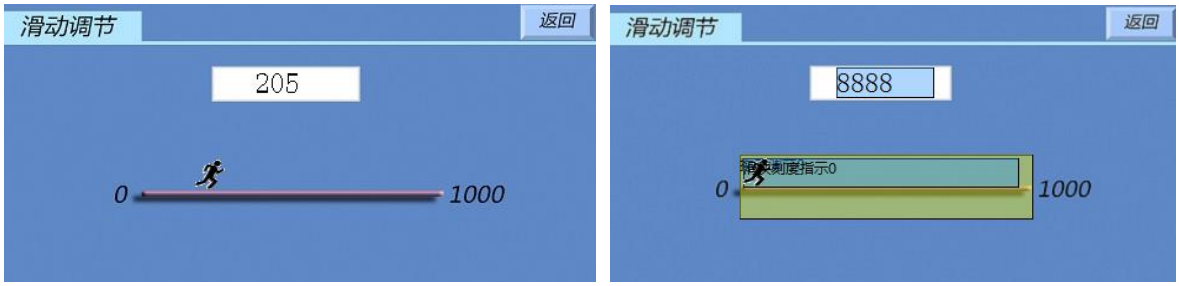



图5-4-4 拖动调节举例

如图 5-4-4 左图，点击并拖动人形滑块，人形滑块会跟随移动，同时上方文本框数值会跟随变化。为实现上述典型的拖动功能，如上图右侧，需要设计一个数据变量、一个拖动调节按钮、一个滑块刻度显示控件，并且三者必须定义相同的变量存储地址。

例如，如图 5-4-4 设计，拖动调节属性设置如图 5-4-3，拖动“拖动调节”按钮，串口返回指令：

A5 5A 06 83 01 80 01 00 00

……

A5 5A 06 83 01 80 01 03 E8

说明：A5 5A：帧头

06：指令字节长度，83 01 80 01 00 00 共 6 字节（不含帧头）

83：读变量存储器指令

01 80：变量地址，在控件属性中设置

01：数据字长度，00 00 共 1 字长

00 01：数据内容，从设置的起始值变化到终止值，即从 00 00 连续变化到 03 E8

拖动录入的优点是直接、快捷，而且参数不会越界。拖动调节不支持按键控制（不支持外接键盘和 4F 寄存器）。为防止误操作，拖动调节须按压有效拖动区域超过 0.1 秒后拖动才启动。

### 5.4.3 转动调节

和“拖动调节”按钮类似，按下并移动“转动调节”按钮，可以将指定变量、按照设定的范围自动变化。**转动调节不支持按键控制（不支持外接键盘和4F寄存器）。**转动调节始终假定为顺时针转动。

定义按钮位置、大小，以及环形调节区域。**按钮必须包含转动调节环形区域。**

定义转动调节环形区域。

字地址，范围 0000~FFFF，用于存储变量，长度为一个单元（字地址一个单元包含 2 个字节）。按下转动调节按钮后，系统会根据转动调节按钮的当前位置自动计算变量值（根据起始值和终止值进行插值运算）。使用 0x83 指令上传变量时会使用该地址。

调节区域起始角度 0-719，单位为 0.5°

起始角度对应的返回值，整数。

调节区域终止角度 1-720，单位为 0.5°

终止角度对应的返回值，整数。

| 区域范围设置 | |
|---|---|
| X坐标 | 285 |
| Y坐标 | 82 |
| 宽度 | 176 |
| 高度 | 160 |
| 移动锁定 | ☐ |
| 圆心坐标 | 0,0 |
| 区域内径R0 | 1 |
| 区域外径R1 | 1 |
| **按钮属性** | |
| **变量属性** | |
| 变量存储地址(0x) | 0000 |
| 变量模式 | 按字调节 |
| 数据自动上传 | ☑ |
| **控件属性** | |
| 起始角度A0 | 0 |
| 起始角度对应变量值 | 0 |
| 终止角度A1 | 1 |
| 终止角度对应变量值 | 0 |

图5-4-5 转动调节按钮属性设置

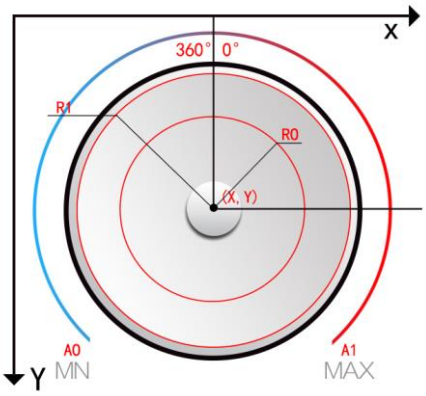



图 5-4-6 转动调节区域示意图

### 5.4.4 硬件参数配置

常用于把变量存储器内的数据写入寄存器里，或者把寄存器内的数据写入变量存储器里。在变量存储器区开辟了一块特殊区域 0x6F00-0x6FFF，共 256 个单元（仅仅使用低字节），这 256 个单元与寄存器区单元（0x00-0xff）一一对应。

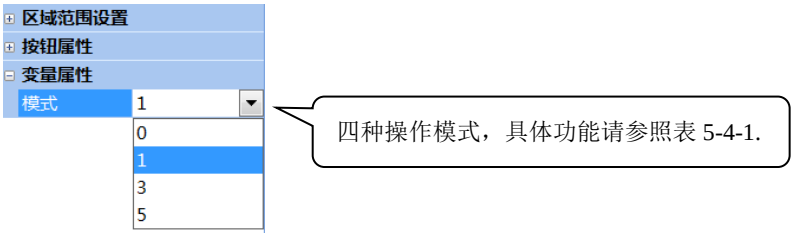



图 5-4-7 硬件参数配置按钮属性设置

表 4-4-1 硬件参数配置按钮操作模式

| 模式 | Data_Pack | Data_Pack说明 | 功能 |
|---|---|---|---|
| 0 | 无 | 无 | 把 256 字节寄存器区数据复制到到变量存储器单元 0x6F00-0x6FFF（占据低字节）。VGUS 屏上自动把寄存器区数据复制到 0x6F00-0x6FFF 单元。 |
| 1 | 无 | 无 | 把变量存储器单元 0x6F00-0x6FFF（占据低字节）数据复制到寄存器区。 |
| 3 | *VP | 数据指针 | 把*VP 指针位置开始的，Tx_Len 字节长度的数据发送到用户串口。Tx_Len 是双字节变量，长度从 0x0001-0xFFFF。 |
| | Tx_LEN | 要发送的数据长度 | |
| 5 | Tran_Area | 将要转换的区域坐标：左上角、右下角 | 将指定区域的内容转换成单色位图（横向取模打印位图格式），并保存到 VP 指针指向的数据存储器。<br>1.区域宽度（Xe-Xs+1）必须是 16 的倍数；<br>2.*VP 指针保存数据格式如下：<br>*VP：状态位，处理完成后设置成 0x5555；<br>*VP+1：横向字长度＝（Xe-Xs＋1）&0xFFF0/16；<br>*VP+2：数据段个数＝（Ye-Ys＋1）；<br>*VP+3：位图数据开始，MSB 方式。<br>如果启用了“参数自动上传功能”（R2.3=1），那么转换完成后，会按照*VP 内容被修改成 0x5555 而自动上传一条提示信息。<br>本指令主要用于屏幕内容的打印输出。 |
| | *VP | 保存转换位图数据的缓冲区首地址 | |
| | Frame_End | 帧尾（两字节） | |

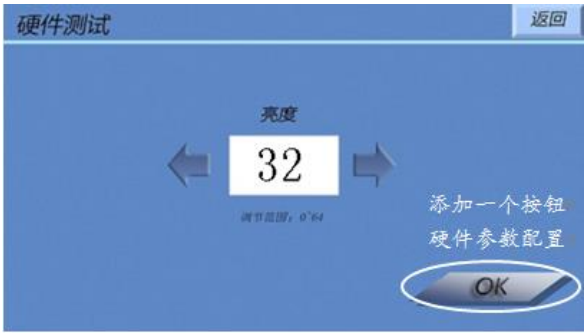



图 5-4-10 硬件参数配置举例

在增量调节图 5-4-2 的例子中，已经有用到上图例子，通过左右键增量调节按钮我们可以修改文本框中的亮度值。如图 5-4-8 中，再在右下角添加一个硬件参数配置按钮“OK”，就可以直接用文本框中的数值去控制液晶屏的亮度，硬件参数配置按钮属性如图 5-4-7。

为实现上述功能，必须将增量调节按钮、数据变量的变量存储器地址都设置为 0x6F01，正好对应亮度设置寄存器 0x01，硬件参数配置按钮设置为模式 1。当硬件参数配置按钮按下后，变量存储器 0x6F00-0x6FFF 单元里的数据（仅低字节）会复制到寄存器区，其中 0x6F01 单元里的亮度值正好对应写入亮度寄存器 0x01 单元，屏幕亮度跟随变化。

如果工程中用到硬件参数配置按钮，则所有变量地址必须避开 0x6F00-0x6FFF 区域。

# 6 描述指针

对于简单应用场合，用户无需了解本章知识。

***SDWn 系列不支持描述指针功能***

## 6.1 描述指针介绍

VGUS 屏采用变量驱动，所有显示的字符、图标等都定义为一个变量，并分配变量存储地址、定义显示格式，然后生成配置文件并下载到显示终端上。在需要刷新显示时，用户仅需将变量内容和变量存储地址通过串口对应发送到显示终端，显示终端会自动按照定义好的显示格式显示。

对于每个变量来讲，其显示格式是固定的，是由下载到 VGUS 屏中的配置文件定义的。当用户需要临时修改变量显示格式时，可以通过本章介绍的描述指针实现。

用户在定义每个变量的时候，需要设置是否启用描述指针，如图 6-1 所示。

图 6-1 设置数据变量显示属性

如图描述指针一栏填写 FFFF 表示当前变量禁止描述指针功能。如果填写的是其它数据（0000-FFFE 中任意一个）表示当前变量启用描述指针，并且填写的数据用来指定变量存储器地址，以该地址为起始单元，在变量存储器内连续开辟了一块空间（不同的变量类型，块长度以及数据格式也都不同，每种变量本章都有详细表格定义了对应数据格式），用于存储变量的显示格式。用户可以通过变量存储器地址，利用 0x82 指令去动态修改变量显示格式。

因为最长的变量属性长度为 32bytes（16Words），为了避免出现不同变量的描述指针重叠，建议一律按照最长的变量属性 16word 分配描述指针地址。例如第一个变量描述指针地址为 0x0100，则第二个变量描述指针地址设置为 0x0110，这样就可以确保不同变量的描述指针地址不会出现重叠。

每一个显示变量的属性都可以通过描述指针来读写。描述指针地址范围 0x0000-0xFFFE，与用户变量共用变量存储器区，因此用户定义的变量存储地址不能与描述指针的地址重叠。

## 6.2 描述指针应用举例

本节以改变一个数据变量的显示颜色为例，介绍描述指针的简单应用。

1、 新建工程添加图片后，在界面上添加一个数据变量显示控件，设置描述指针为 4000，变量存储地址为 0000. 如图 6-2-1 所示；

| | |
|---|---|
| 名称定义 | 数据变量显示 |
| 描述指针(0x) | 4000 |
| 变量存储地址(0x) | 0000 |
| 文本颜色 | 119; 119; 119 |
| 字库位置 | 0 |
| 字体大小 | 16 |
| 对齐方式 | 左对齐 |
| 变量类型 | int ( 2Byte ) |
| 整数位数 | 4 |
| 小数位数 | 0 |
| 显示单位 | |
| 初始值 | 1234 |

图 6-2-1 添加数据变量

2、 查看关于数据变量描述指针的说明；

表 6-2-1 数据变量描述指针数据格式

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 变量指针 |
| 0x01 | X,Y | 4 | 起始显示位置，显示字符串左上角坐标 |
| 0x03 | Color | 2 | 显示颜色 |
| 0x04:H | Lib_ID | 1 | ASCII字库位置 |
| 0x04:L | 字体大小 | 1 | 字符X方向的点阵数 |
| 0x05:H | 对齐方式 | 1 | 0x00=左对齐，0x01=右对齐，0x02=居中 |
| 0x05:L | 整数位数 | 1 | 显示的整数位数（整数位数和小数位数之和不能超过20） |
| 0x06:H | 小数位数 | 1 | 显示的小数位数（整数位数和小数位数之和不能超过20） |
| 0x06:L | 变量数据类型 | 1 | 0x00=整数(两字节)：-32768到32767<br>0x01=长整数(4字节)：-2147483648到2147483647<br>0x02=VP*高字节：0到255<br>0x03=VP*低字节：0到255<br>0x04=超长整数(8字节)：<br>-9223372036854775808 到 9223372036854775807<br>0x05=无符号整数(2字节)：0到65536<br>0x06=无符号长整数(4字节)：0到4294967295 |
| 0x07:H | Len_unit | 1 | 变量单位显示长度，0x00表示不显示单位 |
| 0x07:L | String_unit | 10 | 单位字符串，ASCII编码 |

从图中可以看到，颜色对应的偏移量为03，那么颜色属性对应的变量存储器地址为0x4000+03=0x4003。通过 0x82 指令修改 0x4003 单元内的数据即改变了数据变量的显示颜色。

例如串口发送：A5 5A 05 82 40 03 F8 00

那么就设置数据变量显示为红色（0xF800 为红色）。

本例中使用按钮键值返回控件，按下按钮时，直接将键值作为对应颜色值写入变量存储器 0x4003 单元。按键返回控件属性设置如图 6-2-2 所示。

| 按钮属性 | |
|---|---|
| 名称定义 | 按键返回 |
| 数据自动上传 | ☑ |
| 按钮效果 | 1 |
| 无按钮效果 | ☐ |
| 页面切换 | -1 |
| 无页面切换 | ☑ |
| 语音 | ☐ |
| 键值（0x） | F800 |
| **变量属性** | |
| 变量存储地址(0x) | 4003 |
| 调节模式 | 按字写入变量 |

图 6-2-2 按键返回控件属性设置

## 6.3 描述指针汇总

表 6-3 描述指针汇总表

| 编号 | 名称 |
|---|---|
| 表 6-3-1 | 变量图标 |
| 表 6-3-2 | 动画图标 |
| 表 6-3-3 | 滑动刻度 |
| 表 6-3-4 | 艺术字变量 |
| 表 6-3-5 | 动画图片 |
| 表 6-3-6 | 旋转图标 |
| 表 6-3-7 | 位变量图标 |
| 表 6-3-8 | 数据变量 |
| 表 6-3-9 | 文本变量 |
| 表 6-3-10 | 文本时钟 |
| 表 6-3-11 | 表盘时钟 |
| 表 6-3-12 | 时钟变量 |
| 表 6-3-13 | 滚动文本 |
| 表 6-3-14 | 实时曲线 |
| 表 6-3-15 | 基本图形 |
| 表 6-3-16 | 列表显示 |
| 表 6-3-17 | 二维码 |

表 6-3-1 变量图标描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 变量指针，变量为整数格式。 |
| 0x01 | X,Y | 4 | 变量显示位置，图标左上角坐标位置。 |
| 0x03 | V_Min | 2 | 变量下限，越界不显示。 |
| 0x04 | V_Max | 2 | 变量上限，越界不显示。 |
| 0x05 | Icon_Min | 2 | V_Min对应的图标ID。 |
| 0x06 | Icon_Max | 1 | V_Max对应的图标ID。 |
| 0x07:H | Icon_Lib | 1 | 图标库存储位置。 |
| 0x07:L | Mode | 1 | ICON显示模式。0x00=透明（不显示背景），其他=显示图标背景。 |

表 6-3-2 动画图标描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 初始图标变量指针，变量为双字，低位字保留，高位字无符号数 |
| 0x01 | x,y | 4 | 变量显示位置，图标左上角坐标位置。 |
| 0x03 | 0x0000 | 2 | 固定 |
| 0x04 | V_Stop | 2 | 变量为该值时固定。 |

| | | | |
|---|---|---|---|
| 0x05 | V_Start | 2 | 变量为该值时自动显示动画图标。 |
| 0x06 | Icon_Stop | 2 | 变量为V_Stop时固定显示的图标。 |
| 0x07 | Icon_Start | 2 | 变量为V_Start值时，自动从Icon_ Start到Icon_ End显示图标，形成动画。 |
| 0x08 | Icon_End | 2 | |
| 0x09:H | Icon_Lib | 1 | 图标库存储位置。 |
| 0x09:L | Mode | 1 | ICON显示模式。0x00=透明 |

表 6-3-3 滑动刻度描述指针定义

| 地址偏移量(双字节) | 定义 | 数据长度(字节) | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 变量指针，变量格式由VP_DATA_Mode来决定。 |
| 0x01 | V_begain | 2 | 对应起始变量刻度值。 |
| 0x02 | V_end | 2 | 对应终止变量刻度值。 |
| 0x03 | X_ begain | 2 | 起始刻度坐标（纵向为Y坐标）。 |
| 0x04 | X_ end | 2 | 终止刻度坐标（纵向为Y坐标）。 |
| 0x05 | Icon_ ID | 2 | 滑动刻度块的图标ID。 |
| 0x06 | Y | 2 | 刻度指示图标显示的Y坐标值。 |
| 0x07:H | X_adj | 1 | 刻度指示图标显示的X坐标前移偏移量。 |
| 0x07:L | Mode | 1 | 刻度模式。0x00=横向刻度条，其他=纵向刻度条。 |
| 0x08:H | Icon_Lib | 1 | 图标库存储位置。 |
| 0x08:L | Icon_mode | 1 | ICON显示模式。0x00=透明（不显示背景），其他=显示图标背景。 |
| 0x09:H | VP_DATA_Mode | 1 | 0x00=*VP指向一个整型变量；<br>0x01=*VP指向一个整型变量高字节数据；<br>0x02=*VP指向一个整型变量低字节数据。 |

表 6-3-4 艺术字变量

| 地址偏移量(双字节) | 定义 | 数据长度(字节) | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 变量指针 |
| 0x01 | X,Y | 4 | 起始显示位置<br>左对齐模式，坐标为显示字符串左上角坐标<br>右对齐模式，坐标为显示字符串的右上角坐标 |
| 0x03 | Icon0 | 2 | 0对应的ICON_ID，排列顺序为0123456789-. |
| 0x04:H | Icon_Lib | 1 | 图标库存储位置。 |
| 0x04:L | Icon_Mode | 1 | ICON显示模式：0x00=透明（不显示背景）<br>其他=显示图标背景 |
| 0x05:H | 整数位数 | 1 | 显示的整数位数 |
| 0x05:L | 小数位数 | 1 | 显示的小数位数 |
| 0x06:H | 数据变量类型 | 1 | 0x00=整数(2 字节)，-32768 到 32767<br>0x01=长整数(4 字节)-2147483648 到 2147483647<br>0x02=*VP 高字节,无符号数0 到 255<br>0x03=*VP 低字节，无符号数0 到 255<br>0x04= 超 长 整 数 (8 字 节 )-9223372036854775808 到 9223372036854775807<br>0x05=无符号整数(2 字节)0 到 65535<br>0x06=无符号长整数(4 字节)0 到 4294967295 |
| 0x06:L | 对齐模式 | 1 | 0x00=左对齐，0x01=右对齐 |

表 6-3-5 动画图片描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | 0x0000 | 2 | 固定 |
| 0x01 | Pic_begain | 2 | 起始图标位置 |
| 0x02 | Pic_end | 2 | 终止图标位置 |
| 0x03:H | Frame_Time | 1 | 一幅图片显示的时间，8ms |

表 6-3-6 旋转图标描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 变量指针，变量格式由Mode决定。 |
| 0x01 | Icon_ID | 2 | 指定图标ID。 |
| 0x02 | Icon_Xc | 2 | Icon图标上的指定旋转中心位置X坐标。 |
| 0x03 | Icon_Yc | 2 | Icon图标上的指定旋转中心位置Y坐标。 |
| 0x04 | Xc | 2 | Icon图标显示到当前屏幕的旋转中心位置X坐标。 |
| 0x05 | Yc | 2 | Icon图标显示到当前屏幕的旋转中心位置Y坐标。 |
| 0x06 | V_Begain | 2 | 对应起始旋转角度的变量值，整型，越界不显示。 |
| 0x07 | V_End | 2 | 对应终止旋转角度的变量值，整型，越界不显示。 |
| 0x08 | AL_Begain | 2 | 起始旋转角度0-720（0x0000-0x02D0),单位0.5度。 |
| 0x09 | AL_End | 2 | 转动角度=（终止角度-起始角度）/（终止值-起始值）*输入值；<br>终止旋转角度0-720（0x0000-0x02D0)，单位0.5度。 |
| 0x0A：H | VP_Mode | 1 | 0x00=*VP指向一个整型变量；<br>0x01=*VP指向一个整型变量高字节数据；<br>0x02=*VP指向一个整型变量低字节数据。 |
| 0x0A：L | Lib_ID | 1 | 图标库存储位置。 |
| 0x0B | Mode | 1 | ICON显示模式。0x00=透明（不显示背景），其他=显示图标背景。 |

表 6-3-7 位变量图标

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 位变量指针，字变量 |
| 0x01 | *VP_AUX | | 辅助变量指针，双字，用户禁用 |
| 0x02 | Act_Bit_Set | 2 | 为1的bit值说明*VP对应位置需要显示 |
| 0x03：H | Display_Mode | 1 | 显示模式定义：<br>Display_Mode：Bit值 0 / 1 |

| | | | |
|---|---|---|---|
| | | | 0x00 ICON0S ICON1S<br>0x01 ICON0S 不显示<br>0x02 ICON0S ICON1S-ICON1E动画<br>0x03 不显示 ICON1S<br>0x04 不显示 ICON1S-ICON1E动画<br>0x05 ICON0S-ICON0E动画 ICON1S<br>0x06 ICON0S-ICON0E动画 不显示<br>0x07 ICON0S-ICON0E动画 ICON1S-ICON1E动画 |
| 0x03：L | Move_mode | 1 | 位图图标排列方式：<br>0x00=X++,Act_Bit_Set指定的不处理bit不保留位置<br>0x01=Y++,Act_Bit_Set指定的不处理bit不保留位置<br>0x02=X++,Act_Bit_Set指定的不处理bit保留Dis_MOV位置<br>0x03=Y++,Act_Bit_Set指定的不处理bit保留Dis_MOV位置 |
| 0x04:H | Icon_Mode | 1 | ICON显示模式：0x00=透明（不显示背景）<br>其他=显示图标背景 |
| 0x04:L | Icon_Lib | 1 | 图标库存储位置 |
| 0x05 | ICON0S | 2 | 不显示动画模式，bit 0 图标 ID<br>显示动画模式，bit 0 图标动画起始 ID 位置 |
| 0x06 | ICON0E | 2 | 显示动画模式，bit 0 图标动画结束 ID 位置 |
| 0x07 | ICON1S | 2 | 不显示动画模式，bit 1 图标 ID<br>显示动画模式，bit 1 图标动画起始 ID 位置 |
| 0x08 | ICON1E | 2 | 显示动画模式，bit 1 图标动画结束 ID 位置 |
| 0x09 | (x,y) | 4 | 起始变量显示位置，图标左上角坐标 |
| 0x0B | Dis_MOV | 2 | 下一个图标坐标移动的间隔 |

表 6-3-8 数据变量描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 变量指针 |
| 0x01 | X,Y | 4 | 起始显示位置，显示字符串左上角坐标 |
| 0x03 | Color | 2 | 显示颜色 |
| 0x04:H | Lib_ID | 1 | ASCII字库位置 |
| 0x04:L | 字体大小 | 1 | 字符X方向的点阵数 |
| 0x05:H | 对齐方式 | 1 | 0x00=左对齐，0x01=右对齐，0x02=居中 |
| 0x05:L | 整数位数 | 1 | 显示的整数位数（整数位数和小数位数之和不能超过20） |
| 0x06:H | 小数位数 | 1 | 显示的小数位数（整数位数和小数位数之和不能超过20） |
| 0x06:L | 变量数据类型 | 1 | 0x00=整数(两字节)：-32768到32767<br>0x01=长整数(4字节)：-2147483648到2147483647 |

| | | | |
|---|---|---|---|
| | | | 0x02=VP*高字节：0到255<br>0x03=VP*低字节：0到255<br>0x04=超长整数(8字节)：<br>-9223372036854775808 到 9223372036854775807<br>0x05=无符号整数(2字节)：0到65536<br>0x06=无符号长整数(4字节)：0到4294967295 |
| 0x07:H | Len_unit | 1 | 变量单位显示长度，0x00表示不显示单位 |
| 0x07:L | String_unit | 10 | 单位字符串，ASCII编码 |

表 6-3-9 文本变量描述指针定义

| 地址偏移量(双字节) | 定义 | 数据长度(字节) | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 文本指针 |
| 0x01 | X,Y | 4 | 起始显示位置，显示字符串左上角坐标 |
| 0x03 | Color | 2 | 显示文本颜色 |
| 0x04 | Xs,Ys,Xe,Ye | 8 | 文本框 |
| 0x08 | Text_Length | 2 | 显示字节数量，遇到0xFFFF数据或者显示到文本框尾则不再显示 |
| 0x09:H | Font0_ID | 1 | 编码方式为0x01-0x04时ASCII字库位置 |
| 0x09:L | Font1_ID | 1 | 编码方式为0x00、0x05,以及0x01-ox04的非ASCII字符使用的字库位置 |
| 0x0A:H | Font_X_Dots | 1 | 字体X方向点阵数，0x01-0x04模式时ASCII字符X按照Y/2计算 |
| 0x0A:L | Font_Y_Dots | 1 | 字体Y方向点阵数，字库中Y方向的点阵数必须为偶数 |
| 0x0B:H | Encode_Mode | 1 | .7定义了文本显示字符间距是否自动调整：<br>.7=0字符间距自动调整<br>.7=1字符间距不自动调整，字符宽度为固定设定的点阵数<br>.6-.0定义了文本的编码方式：<br>0x00=8bit编码，0x01=GB2312内码，0x02=GBK，0x03=BIG5，0x04=SJIS，<br>0x05=UNICODE |
| 0x0B:L | HOR_Dis | 1 | 字符水平间距 |
| 0x0C:H | VER_Dis | 1 | 字符垂直间距 |
| 0x0C:L | 未定义 | 1 | 写0x00 |

表 6-3-10 文本时钟描述指针定义

| 地址偏移量(双字节) | 定义 | 数据长度(字节) | 说明 |
|---|---|---|---|
| 0x00 | 0x0000 | 2 | |
| 0x01 | X,Y | 4 | 起始显示位置，显示字符串左上角坐标 |
| 0x03 | Color | 2 | 字体颜色 |

| | | | |
|---|---|---|---|
| 0x04:H | Lib_ID | 1 | 字库位置 |
| 0x04:L | 字体大小 | 1 | X方向的点阵数 |
| 0x05 | String_Code | Max14 | 编码字符串，使用RTC编码表和ASCII字符组成<br>假设当前时间2014-05-01 12:00:00 星期三<br>Y_M_D H:Q:S 0x00将显示为 2014-05-01 12:00:00<br>M_D W H:Q 0x00将显示为 05-01 WEN 12:00 |

RTC 编码表

| 说明 | 编码 | 显示格式 |
|---|---|---|
| 公历_年 | Y | 2000-2099 |
| 公历_月 | M | 01-12 |
| 公历_日 | D | 01-31 |
| 公历_小时 | H | 00-23 |
| 公历_分钟 | Q | 00-59 |
| 公历_秒 | S | 00-59 |
| 公历_星期 | W | SUN MON TUE WED THU FRI SAT |
| 编码结束 | 0x00 | |

表 6-3-11 表盘时钟描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | 0x0001 | 2 | |
| 0x01 | X,Y | 4 | 时钟表盘的指针中心 |
| 0x03 | Icon_Hour | 2 | 时针 ICON 的 ID，0xFFFF 表示时针不显示。 |
| 0x04 | Icon_Hour_central | 4 | 时针 ICON 的旋转中心位置。 |
| 0x06 | Icon_Minute | 2 | 分针 ICON 的 ID，0xFFFF 表示分针不显示 |
| 0x07 | Icon_Minute_central | 4 | 分针 ICON 的旋转中心位置。 |
| 0x09 | Icon_Second | 2 | 秒针 ICON 的 ID，0xFFFF 表示秒钟指针不显示 |
| 0x0A | Icon_Second_central | 4 | 秒针 ICON 的旋转中心位置。 |
| 0x0C:H | ICON_Lib | 1 | 指针图标所在的 ICON 库文件 ID |
| | 未定义 | 1 | 写 0x00 |

表 6-3-12 变量时钟描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 0x00 | *VP | 2 | 变量指针数据串首地址，变量为BCD编码 |
| 0x01 | X,Y | 4 | 起始显示位置，显示字符串左上角坐标 |
| 0x03 | Color | 2 | 字体颜色 |
| 0x04:H | Byte_Num | 1 | *VP指针高字节开始显示的数目，0x01-0x0F |
| 0x04:L | Lib_ID | 1 | 字库位置，必须是半角字库，如果Lib_ID不为0，则必须用8bit编码 |
| 0x05:H | Font_X | 1 | X方向点阵数目 |
| 0x05:L | String_Code | Max13 | 编码字符串，用来和时间变量组合出客户需要的显示格式，每显示一个BCD编码后，会从编码字符串中顺序的取出一个ASCII字符来间隔显示<br>编码字符串中特殊编码定义：<br>0x00：无效，不显示字符，两个BCD编码连在一起<br>0x0D：换行显示。即 X=Xs，Y=Y+Font_X*2 |

表 6-3-13 滚动文本描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 文本指针<br>文本指针前 3 个字必须保留，用户显示文本内容从（VP+3）开始存放，文本必须以 0xFF 或 0x00 结尾 |
| 0x01:H | Rolling_Mode | 1 | 滚屏模式：0x00表示从右向左滚屏 |
| 0x01:L | Rolling_Dis | 1 | 滚屏间距，每个VGUS周期文本滚动的像素点阵数 |
| 0x02:H | Adjust_Mode | 1 | 对齐方式：0x00=左对齐，0x01=右对齐，0x02=居中<br>文本显示内容不足文本框时滚屏停止，此时显示对齐模式方有效 |
| 0x02:L | 未定义 | 1 | 写 0x00 |
| 0x03 | Color | 2 | 显示文本颜色 |
| 0x04 | Xs Ys Xe Ye | 8 | 文本框 |
| 0x08:H | Font0_ID | 1 | 编码方式为 0x01-0x04 时：ASCII 字符显示的字库位置<br>编码方式为 0x00、0x05 时：本参数不用设置，写 0x00 即可 |
| 0x08:L | Font1_ID | 1 | 编码方式为 0x01-0x04 时：非ASCII字符显示的字库位置<br>编码方式为 0x00、0x05 时：显示字符使用的字库位置 |
| 0x09:H | Font_X_Dots | 1 | 字体X方向点阵数（0x01-0x04模式，ASCII字符X将自动按照 Y/2计算） |
| 0x09:L | Font_Y_Dots | 1 | 字体Y方向点阵数目 |
| 0x0A:H | Encode_Mode | 1 | .7 定义了文本显示的字符间距是否自动调整：<br>.7=0 字符间距自动调整；<br>.7=1 字符间距不自动调整，字符宽度固定为设定的点阵数。<br>.6-.0 定义了文本编码方式：<br>0=8bit编码，1=GB2312内码，2=GBK，3=BIG5，4=SJIS，5=UNICODE |
| 0x0A:L | Text_Dis | 1 | 字符间隔 |

表 6-3-14 实时曲线描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | 0x0000 | 2 | 无定义 |
| 0x01 | Xs:Ys:Xe:Ye | 8 | 曲线窗口：左上角坐标（Xs,Ys），右下角坐标（Xe,Ye） |
| 0x05 | Y_Central | 2 | 曲线中心位置 |
| 0x06 | VD_Central | 2 | 中心曲线对应的曲线数据值 |
| 0x07 | Color | 2 | 曲线颜色 |
| 0x08 | NUL_Y | 2 | 曲线放大倍数，单位是1/256，0x0000-0xFFFF |
| 0x09:H | CHANEL | 1 | 数据源通道：0x01-0x07 |
| 0x09:L | Dis_HOR | 1 | 横轴间隔：0x01-0xFF |

表 6-3-15 基本图形描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 变量数据指针 |
| 0x01 | Area | 8 | 绘图显示区域定义：指定区域的左上角、右下角坐标<br>绘图越界将不显示，仅对 0x0001-0x0005、0x0009、0x000A、0x000B 指令有效 |
| 0x05:H | Dashed_Line_Ent | 1 | 0x5A：使用线段的绘图指令（0x02、0x03、0x09、0x0A 指令）将使用虚线或者点划线显示线段<br>其它：使用线段的绘图指令使用实线显示线段 |
| 0x05:L | Dash_Set | 4 | 4 个字节依次设置了虚线（点划线）格式：<br>第 1 段实线点阵数、第 1 段虚线点阵数、第 2 段实线点阵数、第 2段虚线点阵数<br>例如，设置 0x10 0x04 0x10 0x04 将显示虚线，设置 0x10 0x04 0x02 0x04 将显示点划线 |
| | | 13 | 保留，写0x00 |

表 6-3-16 列表显示描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 表格内容指针，即TAB_X_Num与TAB_Y_Num数组的首地址 |
| 0x01:H | TAB_X_Num | 1 | 列数目，0x01-0xFF |
| 0x01:L | TAB_Y_Num | 1 | 行数目，0x01-0xFF |
| 0x02:H | TAB_X_Start | 1 | 表格起始显示的列位置，0x00-0xFF |
| 0x02:L | TAB_Y_ Start | 1 | 表格起始显示的行位置，0x00-0xFF |
| 0x03:H | Unit_Data_Num | 1 | ➢ 0x01-0xFF所有单元格存储的数据长度相同，<br>➢ 0x00由*VP变量地址指针指向变量存储空间定义不同列单元格的数据长度。（word,字长度） |

| | | | |
|---|---|---|---|
| | | | 当 Unit_Data_Num=0x00 时，表格数据内容存储位置相应后延（TAB_X_Num/2）向上取整个字地址<br>例如，*VP=0x1000，TAB_X_Num=0x07，那么：<br>0x1000-0x1003 依次存储了第 0-6 列的表格数据长度，其中 1003的低字节未使用。0x1004 地址开始存储表格内容 |
| 0x03:L | Encode_Mode | 1 | .7定义了文本显示字符间距是否自动调整：<br>.7=0字符间距自动调整<br>.7=1字符间距不自动调整，字符宽度为固定设定的点阵数<br>.6定义了表格内容格式：<br>.6=0 表格内容为文本格式<br>.6=1.未定义<br>.5 定义了边框线条是否显示：.5=0 显示边框，.5=1 不显示边框<br>.4 未定义，写 0。<br>.3-.0定义了文本的编码方式:0x00=8bit编码,0x01=GB2312内码,0x02=GBK,0x03=BIG5，0x04=SJIS，0x05=UNICODE |
| 0x04 | Xs,Ys,Xe,Ye | 8 | 表格显示区域定义，左上角坐标，右下角坐标，越界不显示 |
| 0x08 | Color_line | 2 | 表格边框颜色 |
| 0x09 | Color_text | 2 | 表格文本颜色 |
| 0x0A:H | Font0_ID | 1 | 编码方式为0x01-0x04时ASCII字库位置 |
| 0x0A:L | Font1_ID | 1 | 编码方式为0x00、0x05,以及0x01-0x04的非ASCII字符使用的字库位置 |
| 0x0B:H | Font_X_Dots | 1 | 字体X方向点阵数，0x01-0x04模式时ASCII字符X按照X/2计算 |
| 0x0B:L | Font_Y_Dots | 1 | 字体Y方向点阵数，字库中Y方向的点阵数必须为偶数 |
| 0x0C:H | TAB_X_Adj_Mod | 1 | 设置TAB_X_Start不为0时，进行表头显示控制，0x00=首列不显示，0x01=首列显示 |
| 0x0C:L | TAB_Y_Adj_Mod | 1 | 设置TAB_Y_Start不为0时，进行表头显示控制，0x00=首行不显示，0x01=首行显示 |

备注[1]：当 Encode_mode.6=1 时，每个单元格数据内容的前两个字定义了表格数据格式，说明如下：

第1个字高字节：Mode 选择数据类型

0x00=整数(2 字节)，-32768 到 32767

0x01=长整数(4 字节)-2147483648 到 2147483647

0x02=*VP 高字节,无符号数0 到 255

0x03=*VP 低字节，无符号数0 到 255

0x04=超长整数(8 字节) -9223372036854775808 到 9223372036854775807

0x05=无符号整数(2 字节)0 到 65535

0x06=无符号长整数(4 字节)0 到 4294967295

0x10=时间格式 1，12:34:56BCD 码串

0x11=时间格式 2，12-34-56BCD 码串

0x12=时间格式 3，YYYY-MM-DD HH:MM:SS BCD 码串

0xFF=文本格式

第1个字低字节：

Mode=0x00-0x06定义了变量数据的定点显示格式，高 4bit 表示整数位数，低 4bit 表示小数位数

Mode=0x10-0X11时间 BCD 码串的字节长度

Mode=其它无定义

第 2 个字：定义单元格文本颜色

如果表格实际内容短于 Unit_Data_Num 规定的长度时，使用 0xFFFF 做为单元格文本结束符

对于特别大的表格，通过触摸屏修改 TAB_X_Start、TAB_Y_Start 值可以很方便的实现表格的定位和拖动

表 6-3-17 二维码描述指针定义

| 地址偏移量（双字节） | 定义 | 数据长度（字节） | 说明 |
|---|---|---|---|
| 0x00 | *VP | 2 | 二维码显示内容指针。<br>二维码内容最长 458Bytes,0x0000 或 0xffff 为结束符。 |
| 0x01 | （x，y） | 4 | 二维码显示的坐标位置。<br>（x，y）为二维码左上角在屏幕的坐标位置。<br>二维码图形有 45*45 单元像素（数据少于 155 字节）和 73*73 单元像素（数据少于 459 字节）两种。 |
| 0x03 | Unit_Pixels | 2 | 每个二维码单元像素所占的物理像素点阵大小，0x01-0x07。<br>设置 Unit_Pixels=4，那么每个单元像素将显示为 4*4 点阵大小。 |

# 7 VGUS 开发工具使用说明

本章内容提供有配套的视频教程“VGUS 开发工具及设计流程”。

VGUS 开发工具是一款组态型、图形用户界面设计软件。软件功能强大且易学，一站式管理，用户无需直接操作繁琐的配置文件。

VGUS 开发工具免安装，可直接运行。支持 Windows XP、Win7、Win8 和 Win10 操作系统。Win8 或 Win10 系统下需要设置兼容性才能运行。首次运行 VGUS 开发工具，软件弹出一个窗口，提示设置兼容性的方法，Win8 和 Win10 系统务必按照提示正确设置兼容性。

## 7.1 操作界面

VGUS 开发工具界面如图 7-1-1 所示。

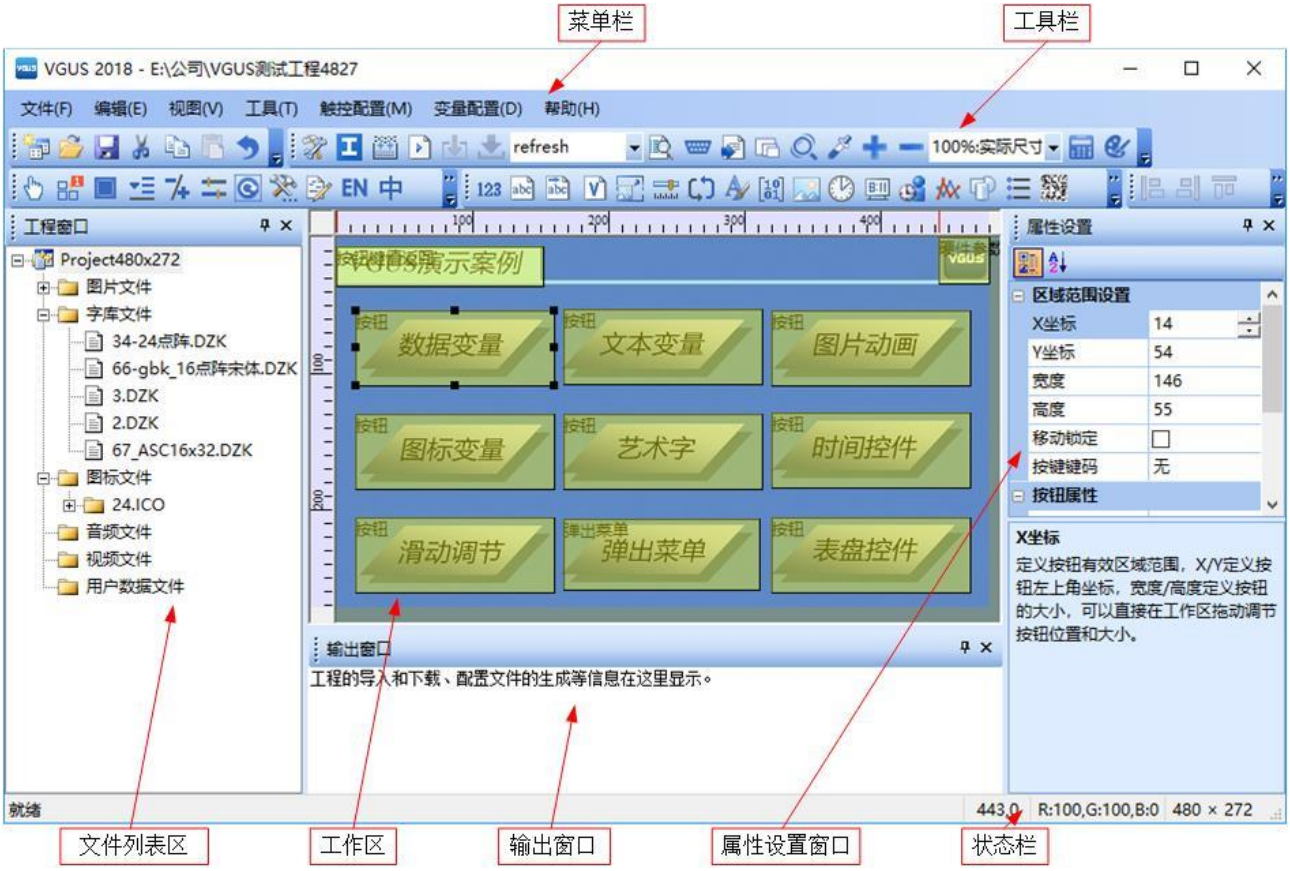



图 7-1-1 VGUS 开发工具软件界面

➢ **菜单栏**

- 文件里面的新建工程经常用到，里面的屏幕属性设置非常重要。
- 编辑菜单功能提供了复制粘贴等常规功能和针对控件的位置大小等编辑功能。
- 视图菜单用于选择工具栏、各窗口是否显示，显示的项目前面会有标记。
- 工具菜单包含了 VGUS 中提供的各类小工具软件，功能如表 7-1-1 所示。屏参配置、生成配置文件、虚拟串口屏、下载到 U 盘、串口调试工具、项目属性设置等都是经常用到的功能。

表 7-1-1 工具汇总

| 工具名称 | 功能说明 |
|---|---|
| 屏参配置 | 用于设置串口波特率、指令帧头、CRC 校验、以及外设和屏保等特性，详见 7.2 节 |
| Icon 生成工具 | 将用户准备好的图标素材转换成 Icon 格式的图标文件 |
| 生成配置文件 | 界面设计完毕后，生成下载文件夹 |
| 虚拟串口屏 | 预览和虚拟调试设计效果，参见文档“虚拟串口屏使用说明” |
| 在线下载 | 通过 USBmini 接口，把下载文件夹下载到串口屏里面，适合研发调试使用 |
| 下载到 U 盘 | 把下载文件夹直接复制到 U 盘，用于脱机下载，适合批量生产使用 |
| 变量查看 | 查看所有控件的地址分配情况 |
| 串口调试工具 | 类似串口调试助手的功能，可以直接生成指令，调试串口屏更方便 |
| 导入工程 | 可以将 VGUS3.2 的工程、其它厂商 xGUS 类的下载文件夹导入生成新工程 |
| 项目属性设置 | 变换产品系列、修改工程分辨率、工程旋转 180 度等 |
| 查找 | 查找变量 |
| 16 位颜色数据 | 将指定颜色转换为 5-6-5 格式数据 |
| 放大 | 调整工作区大小 |
| 缩小 | 调整工作区大小 |
| 计算器 | 打开计算器 |
| 图像编辑工具 | 打开画图 |

- 触控配置用于设计各种触摸按钮控件，详细内容见第 5 章。
- 变量配置用于设计各种显示控件，详细内容见第 4 章。
- 帮助用于可查看当前软件版本。当打开一个工程时，可查看生成该工程所使用的软件版本。

➢ **工具栏**

- 提供了常用菜单的图标快捷方式，鼠标短暂停留可显示文字提示。

➢ **文件列表区**

- 非常重要，用于管理界面中用到的各种素材文件。通过鼠标右键添加、删除文件。

➢ **工作区**

- 用户在此区域进行图形化界面设计，可以拖放移动控件等。
- 点击工作区非控件的地方，属性设置窗口会出现全局设置属性，可以设置自动保存时间间隔。在设计过程中，为防止因意外因素造成的数据丢失，请设置自动保存，或者及时手动保存文件。

➢ **属性设置窗口**

- 用于显示选中控件的属性，不同的控件对应不同的属性，可以对其查看修改；
- 点击某项属性，属性栏下方将显示相应的帮助内容。

➢ **输出窗口：**用于显示工程导入、配置文件生成、下载等操作过程信息。

➢ **状态栏：**用于显示鼠标所指工作区像素点的坐标和颜色信息，以及项目分辨率。

## 7.2 屏参配置

点击菜单栏上“工具->屏参配置”，弹出屏参配置窗口，如图 7-2-1 所示。

图 7-2-1 屏参配置

- **串口参数配置：**
  - 设置串口波特率，VGUS 屏出厂波特率为 115200bps。
  - 设置指令帧头，指令帧头的设置主要达到以下两个目的：（1）用于串口指令帧的识别和同步；（2）多台VGUS屏并联工作时，把帧头作为设备地址加以区分。VGUS屏出厂帧头为A5 5A。
  - 设置是否启用 CRC 校验。
- **参数配置：**
  - 设置变量初值，可以全部设置为 0、或者启用变量属性设置的初始值。
  - 背光设置（启用屏保）。

    设置是否启用屏保，以及屏保延时动作时间、屏保亮度、和正常显示亮度，如表 7-2-1；

    当屏幕进入屏保后，第一次点击触摸屏不会触发动作，而只是恢复进入正常显示状态；

    利用点亮亮度寄存器，也可以实现设置串口屏的默认开机亮度。

表7-2-1 屏保功能寄存器

| 名称 | 取值范围 | 说明 |
|---|---|---|
| R6 | 0x00-0x40 | 触摸屏控制背光启动后，点击触摸屏后背光点亮的亮度。 |
| R7 | 0x00-0x40 | 触摸屏控制背光启动后，一段时间不点击触摸屏，背光关闭的亮度。（屏保保护亮度） |
| R8 | 0x01-0xFF | 触摸屏控制背光启动后，触摸屏背光点亮时间，单位为 1.0 秒。（屏保延时时间） |

如设置启用屏保、R6=0x40、R7=0x10、R8=0x1E，当 30 秒（0x1E）不点击触摸屏，背光亮度将自动降低到 0x10（25%亮度）；点击触摸屏后，背光亮度将自动调节到 0x40（100%亮度）。

  - 设置 VGUS 运行周期。页面上的变量采用定时刷新，建议将固定的背景提示语、单位提示符、表格等直接做到图片上，减少串口屏的刷新任务，有利于提高串口屏的刷新速度。

- 设置 U 盘下载后启动触摸校准，通过该功能可以很方便的实现设备出厂前校准触摸屏。
- 设置触屏录入参数后自动上传到串口。
- 设置关闭蜂鸣器。
- 设置开机页面的图片序号。
- 设置读指令的返回帧延时时间，单位为毫秒。
- 设置 miscellaneous 选项，共 8 个，其中各个位定义如表 7-2-2。

表7-2-2 miscellaneous选项位功能定义（0代表对应位不勾选，1代表勾选）

| 位 | 功能说明 |
|---|---|
| .7 | 未定义 |
| .6-.5 | 修改触摸屏类型（工程与系统参数一起下载时，工程高优先级）<br>00：不修改触摸屏类型<br>01：修改为电阻触摸屏（建议同时开启“U 盘下载配置文件后启动触摸校准”功能）<br>10：修改为 I 类电容触摸屏（GT911）<br>11：未定义 |
| .4 | 系统自检寄存器错误代码是否在屏幕上显示<br>0：关闭屏幕显示错误代码信息；<br>1：启用屏幕显示错误代码信息。 |
| .3 | 数据变量录入范围限制控制位<br>0：不考虑小数位限制（默认状态）；<br>1：要考虑小数位限制。<br>例如：变量设置为 2 位整数 1 位小数，范围设置为 1-100，<br>不勾选则实际允许录入范围为 1~100；<br>勾选后因为有一位小数位，则实际允许录入范围是 0.1~10.0。 |
| .2 | 是否启用触摸屏自动校准控制位。<br>自动校准是指 4 秒内快速点击触摸屏的非按钮区域超过 20 次，则进入触摸屏校准模式。<br>0：关闭自动校准（默认状态）；<br>1：打开自动校准。 |
| .1 | 数据录入长度限制控制位<br>0：非循环录入，录入长度达到设置值后禁止录入（默认状态）；<br>1：循环录入，录入长度超出设置值后高位挤出。 |
| .0 | 触控和弹出键盘等像素控制位<br>0：弹出图块在右侧和下侧各少一个像素（默认状态）；<br>1：弹出图块与控件指定大小位置一致。 |

## 7.3 设计流程

详细设计流程如图 7-3-1 所示。

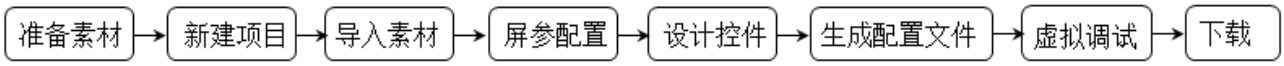



图 7-3-1 详细设计流程

### 1 准备素材

开始使用 VGUS 开发工具设计界面前，首先要准备好用到的背景图片、图标，以及可能用到的字库、音频和视频文件等。文件格式要求参见 7.5 节。

画面中固定显示的文字提醒语、符号、表格和图标等，建议全部做到背景图片上，切不可用变量方式来实现。因为页面上的变量越多，会占用 CPU 的时间越多，影响显示效果。

### 2 新建项目

打开软件，选择“文件→新建”选项，弹出如图 7-3-2 对话框。正确设置产品系列、存储空间、分辨率、旋转角度、项目名称和项目路径。产品系列、存储空间大小、分辨率在产品规格书上都可以查到对应信息。

目前 VGUS 开发工具支持 SDWn/SDWa/SDWe 三个系列串口屏的界面设计，新建工程时要根据实际使用的产品型号正确设置产品系列，否则编译生成下载文件夹后无法下载到串口屏里。

SDWn系列编译生成的下载文件夹为“VT_SET_SDWn”，
SDWa系列编译生成的下载文件夹为“VT_SET_SDWa”，
SDWe系列编译生成的下载文件夹为“VT_SET”。
***必须正确设置产品系列，否则无法正确生产下载文件夹，导致无法下载***

SDWa/SDWn 系列串口屏存储空间有 8M/16M 字节两种，SDWe 系列存储空间一律都是 128M 字节。

旋转角度根据需要设置，如果遇到横屏竖用、或者竖屏横用就需要设置 90 度旋转。注意：设置 90 度旋转时，分辨率不需要交换，还是要参照规格书上给出的分辨率正确设置。例如，SDWa070C03T 这款默认为横屏，规格书给出的分辨率为 800x480，如果需要用作竖屏显示，就要设置 90 度旋转，但是分辨率依然要按照规格书设置为 800x480，而不能设置为 480x800。

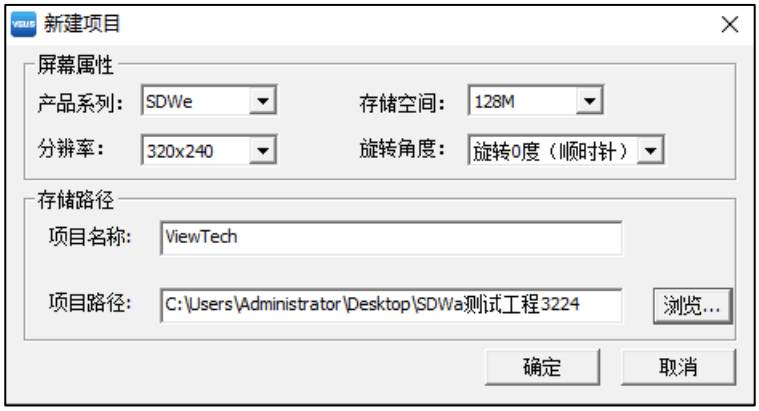



图 7-3-2 新建项目对话框

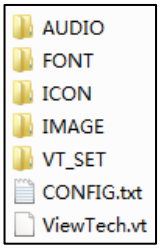



图 7-3-3 项目文件夹

点击图 7-3-2 上确定按钮后，将在项目路径下生成名称与项目名称相同的项目文件夹，如图 7-3-3 所示，用以存放整个项目文件。“*.vt”是 VGUS 项目工程文件，用户可以使用 VGUS 开发工具直接打开。

**3 导入素材**

把准备好的图片、图标、字库、音频文件导入到 VGUS 开发工具左侧的文件列表窗口中。

**4 屏参配置**

根据项目需要，正确设置串口波特率、指令帧头、CRC 校验、蜂鸣器、触摸屏和屏保等外设特性。

**5 设计控件**

包括显示控件（变量配置/第4章）和触摸控件（触控配置/第5章），VGUS开发工具提供有功能完善的多种显示控件和触摸控件，分别见第4章、第5章。在VGUS开发工具菜单栏中或者工具栏上找到相应控件按钮，点击按钮并拖放到工作区，调整控件大小及位置、并在右侧属性窗口中完成控件属性设置。

**6 生成配置文件**

设计完所有显示控件和触摸控件后，点击菜单栏或者工具栏上的“生成配置文件”按钮，会在项目文件夹下生成下载文件夹。输出窗口会提示下载文件夹是否生成成功。产品系列不同，生成的**下载文件夹**名称也不同。SDWa系列产品生成的下载文件夹名称为“VT_SET_SDWa”，SDWe系列产品生成的下载文件夹名称为“VT_SET”。

下载的时候，必须正确拷贝下载文件夹，切不可相互混用。严禁拷贝其他文件到下载文件夹下，也严禁手动删除下载文件夹下任何文件。

**7 虚拟调试**

VGUS开发工具提供有虚拟串口屏和串口调试工具功能。在生成配置文件夹后，可以预览及虚拟调试设计效果（实际效果以下载到VGUS屏内为准）。详细方法参见文档《虚拟串口屏使用方法》。

## 7.4 工程下载

SDWe/SDWa系列串口屏支持通过U盘脱机下载，SDWn系列支持TF卡脱机下载，U盘和TF卡容量大小没有要求，必须格式化为FAT32或者FAT文件格式。

下载需要经过 2 个步骤：

- ✓ 首先是把下载文件夹复制到 U 盘/TF 卡； 把 U 盘/TF 卡插入电脑，刷新盘符并选择插入的 U 盘/TF 卡，点击下载到 U 盘，就可以直接把下载文件夹拷贝到 U 盘/TF 卡里，跳过复制、粘贴的繁琐过程，如下图。

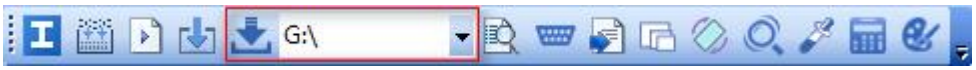



图7-4-1 下载到U盘按钮

根据屏幕属性设置中产品系列设置的不同，会自动识别拷贝下载文件夹 VT_SET、VT_SET_SDWa、VT_SET_SDWn。

- ✓ 然后把装有下载文件夹的 U 盘/TF 卡插入到串口屏上，串口屏将自动启动数据拷贝。拷贝完成后，蜂鸣器会提示下载完成，显示屏左上角提示“Please plug out the U disk.”。 同时会报告成功下载文件数量，下载失败文件数量等信息。待下载完毕后拔出 U 盘即可。

说明：

- ✓ 如果出现下载异常：

  首先排查 U 盘/TF 卡是否有格式化为 FAT32 或者 FAT 文件格式；

  其次排查 U 盘/TF 卡里面是否混入不同产品类型的工程文件；

  然后排查屏幕属性设置是否正确，检查产品系列和分辨率设置是否与所用串口屏一致，可以在“工具-》项目属性设置”中重新设置产品系列、分辨率等屏幕属性；产品系列选择不同，编译生成的下载文件夹不同，SDWe/SDWa/SDWn 系列对应产生的下载文件夹分别为：“VT_SET”、“VT_SET_SDWa” 、“VT_SET_SDWn”。必须正确选择产品系列，产生对应下载文件夹，并将其复制到 U 盘/TF 卡里，否则无法下载。

  如果以上都没有问题，可以尝试更换一只 U 盘/TF 卡。

- ✓ 下载文件夹“VT_SET”、“VT_SET_SDWa” 、“VT_SET_SDWn”下包含了所有的配置文件、字库、图片、图标、音视频文件等，用户不能对该文件夹内文件做任何修改。
- ✓ SDWe 系列串口屏下载文件夹名称用户可以在“VT_SET”后面追加字符如“VT_SETxxxx”。当文件夹名称为“VT_SET_REPLACE”时，下载时只更新同名文件，不删除其它工程文件。该功能适用于终端用户自行替换开机画面。
- ✓ SDWa系列串口屏下载文件夹名称用户可以在“VT_SET_SDWa”后面追加字符如“VT_SET_SDWaxxxx”。SDWa系列串口屏不支持替换下载功能。
- ✓ SDWn系列串口屏下载文件夹名称用户可以在“VT_SET_SDWn”后面追加字符如“VT_SET_SDWnxxxx”。SDWn系列串口屏不支持替换下载功能。

## 7.5 文件格式

VUGS屏支持的文件格式如下表所示。

表7-5-1 VGUS 屏支持的文件格式

| 文件格式 | 含义 | 编号限制 |
|---|---|---|
| *.Jpg | 图片文件，推荐 | 0-65535 |
| *.bmp | 图片文件，不推荐 | 0-65535 |
| *.ico | 图标文件 | 0-255 |
| *.dzk/*.hzk | 用户字库文件 | 0-255 |
| *.mp3 | 音频文件，推荐 | 0-4095 |
| *.wav | 音频文件，不推荐 | 0-4095 |
| *.avi | 视频文件 | 0-65535 |
| *.bin | 用户数据文件 | 0-255 |
| UserDb.bin | 用户数据库文件 | 固定 |

### 1 文件命名规则

上表中所有类型文件的文件名必须以阿拉伯数字开头。例如，要把一副图片序号编为20，图片文件可以命名为“20_测试.jpg”或者“20.jpg”，但不能命名为“测试 20.jpg”。

### 2 图片文件格式

图片的大小要和屏的分辨率一致，液晶屏分辨率在产品规格书中都可以查到，如果下载与屏分辨率不匹配的图片会导致显示异常。所有图片要以数据命名（编号），上电默认显示0号图片，用户也可以在屏参配置中自定义设置。通过寄存器0x003-0x04以0x80指令直接调用、或者通过触摸按钮切换图片显示。

VGUS 开发工具中，图片文件以树状形式管理，如图 7-5-1 所示。通过鼠标右键可以实现图片的添加、删除、排序等操作。

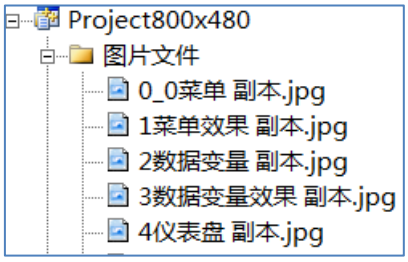



图 7-5-1 图片的管理

VGUS屏支持的图片文件格式为JPG和BMP，如果用户导入其它格式的图片文件，系统将自动转换为JPG格式。强烈建议用户直接使用JPG格式，因为JPG格式占用空间小，显示速度快。JPG格式种类繁多，VGUS屏仅支持Baseline 模式。当使用其它模式的JPG图片后，可能会导致显示异常！

不要用画图、QQ截图等工具直接另存JPG格式文件，这样显示效果较差，因为JPG格式文件为有损压缩文件。建议首先用使用画图、QQ截图等工具保存BMP格式文件，再用PhotoShop工具打开图片，然后另存为JPG格式，具体步骤如下：

1、 用PhotoShop打开图片；

2、 另存为jpg格式；

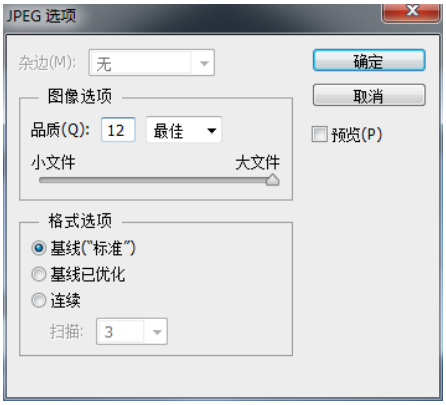



图7-5-2 另存图片品质选项

3、 选择图像效果“最佳”选项。

BMP格式为原始位图文件，包含信息最全，如果使用JPG格式文件显示效果不能满足要求，用户可选择BMP格式文件。使用BMP格式会占用更多的存储空间，画面刷新速度会明显变慢。

## 3 图标文件格式

图标其实就是小图片，所有图标都要以数字命名（编号）。图标支持 bmp、jpg、gif、png 等格式。需要用到图标透明显示场合，图标素材必须使用 BMP 格式，图标左上角像素颜色为背景色。图标素材尺寸建议不要超过 255*255，过大尺寸的图标会影响显示速度。

VGUS 开发工具中，图标以 ICO 文件形式管理，一个 ICO 文件可以包含最多 1024 个图标，如图 7-5-3 所示。一个工程允许存在多个 ICO 文件。ICO 文件新建、添加、删除、浏览等操作都直接通过鼠标右键完成。图标的引用通过 ICO 文件+图标编号的形式，如图 7-5-4 所示。

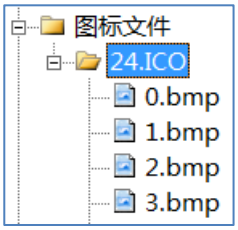



图 7-5-3 图标的管理

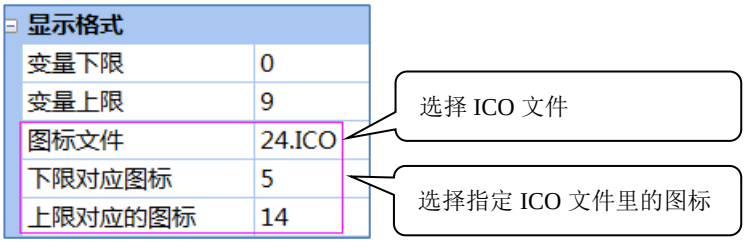



图 7-5-4 图标的引用

ICO文件可以理解为图标子文件夹或者图标集，目的主要为了兼容VGUS4.3开发工具。在VGUS4.3开发工具下需要通过图标生成工具产生ICO文件后，才能添加使用。VGUS2019开发工具提供了更为方便、直观的图标管理形式，又可以直接兼容VGUS4.3的现有设计。

## 4 字库文件格式

VGUS 屏出厂预装有 0 号字库文件（0.dzk），包含 4*8 到 64*128 点阵的所有 ASCII 码字库。

当需要用到其它类型字库时，可以找技术支持索取或者自行通过 TS3 软件生成相应字库文件。新字库需要从 1 开始命名，不要与 0 号字库重名。

TS3 软件及使用说明官网可以下载。字库编码支持 ASCII 码、GB2312 内码、GBK、BIG5、SJIS 以及 UNICODE。相关文档参见《VGUS 串口屏字库编码详解》和《多语种、大容量字库的应用技巧》。

## 5 音频文件格式

支持WAV、MP3两种音频文件格式。支持两种播放方式：扩展指令0x8503和寄存器0x50-0x55，详细信息分别参见本文2.4节、3.3节以及文档《音频播放使用说明及案例》。

WAV格式文件必须选用22KHz、双声道数据。寄存器0x50-0x55播放时单个WAV文件不得超过2M字节，扩展指令8503播放时播放列表的WAV文件总和不得大于2M字节。

MP3格式文件，SDWe系列产品大小没有特殊限制，SDWa系列产品限制为不大于2M字节。

## 6 视频文件格式

支持AVI格式视频文件播放。详细信息参见本文3.4节、4.5节和文档《VGUS串口屏视频播放功能详解》。

注意1：所有文件总和不得大于FLASH可用空间；

注意2：字库、ICON文件、以及WAV文件共同占用一个17M字节空间的临时缓存。

正在使用的字库、ICON图标和WAV文件总和不得大于17M字节。

当缓存不足时，串口屏左上角会提示信息 “Out of memory!”。

字库首次使用时会读入缓存，且永不释放，单个字库文件不得超过17M；

ICON图标文件首次使用时会将指定图标读入缓存，当缓存满时，会释放占用缓存；

WAV文件播放前会读入缓存，播放后释放占用缓存；

# 附 1 系统自检寄存器

系统自检的启动方式：

方式1：读取寄存器0xf3-0xf6，系统自动启动一次所有文件的校验；

方式2：系统执行程序时会实时检查使用到的文件。

系统自检的结构输出方式：

当校验到文件错误信息后，会将错误代码写入寄存器0xf3-0xf6中，并通过以下两种方式输出。

方式1：通过0x81指令用串口发送寄存器0xf3-0xf6中的错误代码；

例如A5 5A 07 81 F3 04 00 51 00 0E

方式2：在屏幕的左上角显示错误代码。

可通过“屏参配置”中的“miscellaneous.4”设置是否显示错误代码，详见图7-2-1和表7-2-2。

勾选“miscellaneous.4”则启用屏幕显示错误代码信息。

如果不希望在最终产品上显示错误代码信息，可以不勾选“miscellaneous.4位”选项。

表附2-1 系统自检错误代码

| (0xf3 0xf4) | 错误代码<br>(0xf5 0xf6) | 描述 |
|---|---|---|
| 0000 | 0x0001 | SPI 初始化异常 |
| 0000 | 0x0002 | 系统参数 sysinitial 异常 |
| 0000 | 0x0003 | 读取 Config 异常 |
| BIN_ID | 0x0004 | 用户数据文件异常 |
| WAV_ID | 0x0005 | Wav 文件异常 |
| Mp3_ID | 0x0006 | Mp3 文件异常 |
| AVI_ID | 0x0007 | AVI 文件异常 |
| FT0_ID | 0x0008 | 0 号字库异常 |
| V22_ID | 0X0009 | 变量初始化文件异常 |
| UNICODE_ID | 0x000a | UNICODE 字库异常 |
| FT_ASC_ID | 0x000b | 用户 ASCII 码字库异常 |
| FT_GB_ID | 0x000c | 用户全角字库异常 |
| ICON_ID[7:0]，<br>ICON_NUM[7:0] | ICON_ID[11:8]，<br>ICON_NUM[11:8]，<br>0x0d | 图标文件异常 |
| PIC_ID | 0x000e | 图片文件异常 |
| 0000 | 0x0010 | 工程校验错误，读 F3 寄存器启动并返回工程自检结果 |
| 0000 | 0x0011 | 串口接收到异常指令 |

# 附 2 SDWe 系列数据加密

SDWe系列部分带有USBmini接口的串口屏，通过USB-mini接口连接到电脑后，实际是被虚拟成硬盘，通过电脑可以直接拷贝串口屏内部数据。因此，当用户设计完成以后，必须对串口屏内的数据进行加密保护，防止拷贝抄袭。**SDWe系列没有USBmini接口的串口屏、以及SDWa等其他系列串口屏，不支持USBmini接口连接电脑，不支持数据导出，永久加密，用户无须考虑数据加密。**

用户首先自行制作一个加密文件“password.txt”，该文件包含一个不超过128字节的任意中英文字符串作为密码，用户需要牢记该密码；然后将密码文件“password.txt”复制到U盘根目录下，当将U盘插到VGUS屏上后，密码将自动拷贝到VGUS屏里，实现对VGUS屏内部数据的加密保护。串口屏一旦加密后，VGUS屏内部数据无法读出，USB-mini口无法再连接电脑。

解密过程与加密相似，首先是将解密文件“de_password.txt”（密码必须与加密文件“password.txt”相同）复制到U盘根目录下，当将U盘插到VGUS屏上后，系统将自动对解密密码进行核对，如果与加密密码一致，VGUS屏将自动解密。解密后，VGUS屏可以重新通过USB-mini接口连接到电脑。

加密文件“password.txt”与解密文件“de_password.txt”内的密码相同，仅仅是文件名不同。密码是一个不超过128字节的任意中英文字符串，必须妥善保存，密码一旦遗失，将无法再次解密。

# 附 3 固件更新

在固件更新过程中，请保证供电稳定！

在固件更新过程中，显示屏会有多次自动重新启动，期间切不可断电！

1. 将固件复制到 U 盘/TF 卡根目录下；
2. 将 U 盘/TF 卡插入串口屏，保证供电稳定，系统自动开始更新固件；

3. 在出现“Please plug out the U disk.”提示信息后（蜂鸣器也会有提示音发出），请拔出 U 盘/TF 卡。

4. 在出现“Update success.”提示信息后，固件更新已经完成，停留 3 秒后自动重启进入工作状态。

**固件与工程一起更新下载的方法：**

目前出货默认固件版本都是2019年初的，对于批量用户需要将固件更新到指定日期的情况，用户可以将指定固件文件放到U盘根目录下，随同用户界面工程一起下载到串口屏里，避免重复下载操作。